IN SITU CHARACTERIZATION OF CORRODING INTERFACES VIA DIGITAL SIGNAL ANALYSIS

By

JOSEPH WARREN HAGER

A DISSERTATION PRESENTED TO THE GRADUATE COUNCIL
OF THE UNIVERSITY OF FLORIDA IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1983

Dedicated to my wife, Jeannie

and to Aric, Kirsten and Anneke.

ACKNOWLEDGMENTS

I am indebted to my wife, Jeannie, whose constant encouragement and personal sacrifice both motivated and, to a large extent, enabled me to carry this program to its conclusion. The opportunity to use the excellent and unique facilities of the Solar Energy Research Institute made my efforts in this area of corrosion research possible. I am particularly grateful to Steve Pohlman and Pat Russel for coordinating my stay at SERI and for their many helpful technical discussions. Bob Fortune developed the software which permits computer control of the signal analyzer and provides data analysis and graphics. He also conceived and designed the optical isolator circuit and provided numerous helpful technical suggestions. Frank Urban and Ron Bagley acted as sounding boards for my ideas and also provided numerous helpful suggestions. Rolf Hummel's interest, encouragement, and representation of my work to the rest of my committee was very much appreciated. To my committee chairman, Ellis Verink, and committee members, John Ambrose and Gerhard Schmid, I say "thank you" for accommodating an unusual mode of graduate study. Finally, I acknowledge the financial support of the U.S. Air Force without which graduate study at this stage in my life would not have been possible.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

Abstract of Dissertation Presented to the Graduate Council
of the University of Florida in Partial Fulfillment
of the Requirements for the Degree of Doctor of Philosophy

IN SITU CHARACTERIZATION OF CORRODING INTERFACES
VIA DIGITAL SIGNAL ANALYSIS

By

Joseph Warren Hager

December 1983

Chairman: Dr. Ellis D. Verink, Jr.
Major Department: Materials Science and Engineering

A technique has been developed for characterizing the dynamic electrochemical behavior of a corroding interface utilizing an off-the-shelf commercially available digital signal analyzer, a micro computer and a conventional potentiostat. By treating the corroding electrode as a relaxed, linear, time-invariant system, it may be modeled as an equivalent network of resistors and capacitors. The assemblage of equipment was successful in experimentally determining the component values in a three-element electronic network simulating a simple electrochemical interface. A white noise voltage signal was applied to the network while simultaneously monitoring the current response. Both time domain signals were converted to the frequency domain via the fast Fourier transform (FFT) and manipulated to yield the network impedance.

Graphical analysis of impedance plots yielded the component values. The technique was then applied to an evaluation of the corrosion of 430 stainless steel in 1N sulfuric acid. The method was successful in determining component values of a 5-element network used to model the electrochemical interface in the active, passive, and transpassive regions of polarization potential and at the corrosion potential. These findings show that electrode impedance measurements made via digital signal analysis are sensitive to the character of a corroding interface. This method, because of its relatively short duration of measurement, and negligible permanent effect on the electrode may become an effective means of monitoring corrosion in situ.

CHAPTER I
INTRODUCTION

What probably began as child-like fascination with the look and feel of metallic lumps in the ashes of an ancient campfire has been retained by man throughout the entire period of his civilization. Indeed, the unique properties of metals have rendered them the most significant class of materials in the technological advancement of man up to the present day. Even now, as significant advances are made in the technology of glasses, ceramics, polymers, and sophisticated composite materials, metals will continue to be exploited for their unique properties.

Accompanying the development of the metallurgical technology to extract and refine metals came the realization that large amounts of energy are required to convert the ore from its native state to a usable metallic form. A more startling realization was that the metals which were won from their ores at such a great expense of energy and effort seemed most eager to return to their native state and did so whenever they were left exposed to the terrestrial environment. As various metals were separated and characterized, they were observed to possess varying degrees of susceptibility to the environmental degradation we call corrosion. Despite whatever other unique properties a metal may possess, its ultimate usefulness and application is very strongly influenced by its resistance to corrosion.

The word "corrosion", although sometimes used to denote the deterioration of any substance in the environment of its use, generally connotes the destruction of a metal in the presence of an aqueous electrolyte or other polar solvent. In the context of the latter definition, the apparent disappearance of the metal may be attributed to an electrochemical reaction of the form:

$$Me \rightarrow Me^{n+} + ne^{-}$$

There are few metals of practical importance for engineering applications which are thermodynamically immune to this anodic reaction in the environment of their intended use. Accepting this fact, the engineer is faced with the problem of determining whether the rate at which this reaction takes place will permit the functional utilization of the device for its design lifetime. In some cases, if the corrosion rate is well established and occurs as a uniform attack, one might simply make the structure thicker than required by strength considerations to allow for material loss by corrosion. Another tack is to select a suitable coating which, in effect, separates the metal from the aggressive environment. In cases where failure of the metal component would result in a catastrophic outcome, the solution is often to select a more expensive, more corrosion-resistant material.

The aforementioned design options are logical courses of action assuming that the rate of the corrosion reaction is known and the reaction occurs uniformly over the entire surface of exposure. Unfortunately, the vast majority of corrosion phenomena are not so straightforward. Some metals, subjected to certain environmental conditions, are observed to deteriorate very rapidly at some locations

while appearing immune at others. Such variations in local behavior may be attributed to local variations in solution concentrations which are related to component configuration, previous cleaning or chemical treatment or mechanical damage to the part. The local variations in corrosion rate might also be due to inhomogeneities in the metal itself introduced during welding or heat treatment; or result from plastic deformation during fabrication; or be caused by local states of stress as the part performs its design function. These observations are particularly disquieting in view of the fact that the local corrosion rate variations may span many orders of magniture--thus rendering an "average" corrosion rate virtually meaningless.

Another group of metals protect themselves by forming a coherent protective film of corrosion product which is stable in the aggressive environment. Local removal of the film results in its immediate reformation since the film consists of the corrosion product. But even such self-protecting metals are susceptible to local variations in behavior. Subjected to differing solution concentrations or inhomogeneities in the underlying metal, the protective film may allow very rapid local corrosion (pitting) which may destroy the function of a metal component without extensive metal loss.

Faced with these observations, the task of making an intelligent assessment of component durability is formidable to say the least. In the attempt to address this task, several strategies have evolved. The first and most straightforward approach is real-time testing of the component or sample coupon in the actual environment of intended use. It suffers from the obvious disadvantages that the time of test precludes a priori assessment of component durability in most cases. There

is also the nagging question of whether the test environment adequately represents the range of conditions that will be experienced by the actual part. A second approach is to expose the component to a more severe environment in an attempt to effect an accelerated test. Considering the non-linear character of all deterioration phenomena, this approach cannot predict absolute durability of a part but may be useful in comparing alternative materials or processing techniques. The third approach is the most scientific and is certainly the most rigorous. In this method, one attempts to determine experimentally and explain theoretically the electrochemical deterioration of the material over the entire range of possible local conditions.

Considerable progress has been made in understanding the mechanism of corrosion processes, via this third approach. For example, in many corrosion reactions, the rate of consumption of electrons in a number of possible cathodic reactions has been found to limit the rate at which the anodic charge transfer step may occur. If the cathodic reaction is hydrogen evolution, the cathodic charge transfer step may be limited by the rate at which the diatomic hydrogen molecules are formed. Since the ultimate goal of corrosion study is to stop or at least inhibit the production of metal ions from solid metal, the large number of interdependent reactions offer, in principle, a wide range of possible points of attacking the corrosion process. On the other hand, the sheer number of possible rate-determining steps also severely complicates the unraveling of corrosion mechanisms.

In the absence of a complete understanding of the corrosion mechanism for a particular metal alloy in a particular environment, one seeks methods of characterizing and classifying the behavior of

corroding systems on a quantitative experimental basis. The determination of corrosion rate by the polarization resistance method represents one such very direct method of characterizing the corroding interface. Unfortunately, the rendition of a corrosion rate by this method is not meaningful in metal systems subject to a localized attack. Such systems often possess passive corrosion product films which protect the metal surface from the environment initially and then degrade catastrophically in small areas. In these cases, one seeks a parameter or combination of parameters which reveals the susceptibility of the passive film to failure. Since the electrochemical interface may be modeled as an equivalent electronic network consisting of resistive and capacitive impedance terms, a method which quantifies these terms would provide a more complete characterization of the corroding interface.

The behavior of an equivalent circuit which contains capacitive elements is a function of the frequency of the applied signal. A complete characterization of the circuit therefore requires a scheme which measures the circuit response to a sequence of single-frequency signal applications. An alternate measurement scheme is to apply a multiple-frequency signal and determine the system transfer function via the fast Fourier transform (FFT) algorithm. Subsequent graphical analysis of the system impedance as a function of frequency can yield quantitative values of electronic components in an assumed network model. Where the complexity of system response precludes a simple network model, the plots produce a characteristic "fingerprint" of the corroding interface.

The data collection period of this latter measurement scheme is obviously shorter than that required for sequential single-frequency measurements. In principle, the measurement time for a multi-frequency analysis is determined by the period of the lowest frequency to be analyzed; the time required to perform the FFT being considerably less than one second. In practice, the data precision is improved by averaging a number of runs which lengthens the measurement period. Furthermore, analysis over several frequency ranges provides higher resolution data at low frequencies. While these practical considerations all lengthen the period of measurements, it is still much shorter (minutes versus hours) than single-frequency sequential collection over the same range of frequencies. It is thus possible, in principle, to completely characterize a corroding interface at reasonably close intervals of time observing changes in network parameters which may, among other things, reflect changes in the susceptibility of passive films to localized breakdown.

Recognizing that a simple electronic network model may not adequately describe the electrochemical behavior of a corroding metal interface, it may not always be practical to attempt to quantify and monitor network parameters. In such cases, one could propose more elaborate models and support hypothetical mechanisms by fitting data to these. From the standpoint of corrosion monitoring, however, one may wish to simply characterize the state of corrosion in an empirical way. Graphical portrayal of data in a particular format might show that certain curve forms correlate with "acceptable" corrosion behavior, for example.

The ultimate purpose of the work described here is to increase the quality and quantity of information which can be obtained from a corroding interface in a given period of time. Since the chosen technique is based on the electrochemical nature of the interface, it can be expected to provide insight into electrochemical reaction mechanisms present under a given set of conditions. Since the technique employs alternating current perturbation signals, it can provide information about the capacitive as well as resistive nature of the corroding interface. Since it utilizes state-of-the-art digital signal analysis equipment, it is able to interrogate the interface in a shorter period of time than required by sequential single-frequency exposure methods.

Individual features of this technique are not unique, having been investigated by Epelboin (1-3), Blanc (4), Creason (5-8), Smith (9-10), Lorenz (11) and others (12-14). However, there has been no known attempt to synthesize the well-known AC electrochemical methods with modern digital signal analysis technology for the specific purpose of investigating corroding interfaces. The consequence of this effort provides the technological basis for advanced automated corrosion monitoring systems and for sophisticated electrochemical interrogation schemes for basic corrosion research.

The specific objectives of the work described herein were (1) to assemble an in situ corrosion monitoring system based on digital signal analysis using off-the-shelf commercially available electronic equipment and (2) to demonstrate capabilities and limitations of such a system in characterizing both quantitatively and empirically the corrosion of 430 stainless steel in 1N sulfuric acid subjected to

imposed potentials between -0.5 and +1.5 v (SCE). The remainder of this dissertation outlines the theoretical basis for impedance modeling and solutions associated with system assembly and generation of the interface transfer function, and demonstrates system capability on simulated and real corroding electrodes.

CHAPTER II
THEORETICAL BASIS FOR IMPEDANCE MODELING

The replacement of steel destroyed by electrochemical corrosion in the terrestrial environment constitutes a significant percentage of the annual production of steel in the United States. But the rusting of steel is only one of a large number of metallic deterioration phenomena which are electrochemical in nature. In fact, all metals are subject to electrochemical deterioration in some aqueous solvents. Even the metals which form protective adherent films are susceptible, as the protective nature of the film changes with environmental conditions. Interest in corrosion and other electrochemical phenomena has led to intense study of the interface between a solid surface on which an electrochemical reaction is taking place and the ionic solution which contacts it.

Electrochemical Interface Models

Double-Layer Capacitance

The essential feature of an electrochemical interface is the presence of an enormous electric field ($\sim 10^7$ V/cm) acting over a region roughly 10 Å in thickness immediately adjacent to the solid surface (15). The arrangement of charges and oriented dipoles in this region was termed the electrical double-layer by Helmholtz (15) and is further explained by Bockris (16). They describe the electrified interface as consisting of two sheets of charge of opposite sign--one in the solid electrode surface and the other in solution. This led to the treatment of the electrified interface as a parallel plate capacitor.

Electrocapillarity data, however, do not entirely support the double-layer model, and this led Gouy (17) and Chapman (18) to propose the so-called diffuse double-layer model which took into account variations in the capacitance with potential. The Gouy-Chapman model asserted that the charge in the electrolyte solution was scattered in thermal disarray rather than being ordered in plate-like fashion immediately adjacent to the solid surface. Stern (19) synthesized the Helmholtz and Gouy-Chapman models stating that some of the charge in the electrolyte is organized in plate-like fashion and some is in thermal disarray. The precise refinements to the theory of the electrified interface do not take away the general conclusion that there is a capacitance associated with the interface.

Faradaic Impedance

In corrosion processes as well as in all other electrochemical phenomena, electrons are transferred between the solid electrode and ions on the solution side of the interface. This is the mechanism by which metal atoms become metal ions and leave the metal surface. According to Faraday's Law, the rate at which metal atoms become metal ions is directly proportional to the rate at which charge is transferred across the interface. Since the rate of charge transfer is defined as current, the corrosion rate is directly proportional to the magnitude of current flow.

The fundamental equation relating the current density across a metal-solution interface is attributed to the work of Butler and Vollmer and is given by:

$$i = i_o \{e^{(1-\beta)\eta F/RT} - e^{-\beta\eta F/RT}\} \qquad (2.1)$$

This equation has the typical form of an Arrhenius rate expression where i_o is a concentration-dependent term called the exchange current density, F is the Faradaic constant, R is the gas constant and β is a symmetry factor. Equation (2.1) shows that the current density is a function of temperature, T, and the polarization pontential, η.

Butler (20) demonstrated that there is a linear relationship between current density and potential at small values of polarization potential, i.e.,

$$i = \frac{i_o \eta F_\eta}{RT} \tag{2.2}$$

For $\beta = 0.5$, $i_o = 1mA/cm^2$ and $T = 25^o$, it can be shown that the error in making the linear approximation does not exceed 5% if the polarization potential is below 30 mv (16). Obviously other values of the given parameters determine other ranges of linearity. Using Ohm's Law, the equivalent electrical impedance to current flow offered by the charge transfer reaction is thus a function of temperature and concentration and may be modeled as a resistor. The term "charge transfer resistance" was coined by Gerischer and Vetter (21).

There are other equivalent electrical impedances in the corrosion process. As reaction between the various atomic species takes place, reactants are consumed and products build in concentration. Transport of reactants to and products from the interface is a diffusion process which may be accelerated by boundary layer thinning caused by convective mass transfer. The case of pure diffusion rate control was treated by Warburg (22) who, after combining the impedances associated with diffusion of both oxidizing and reducing species, found that the

resultant impedance has both real and imaginary elements. The so-called "Warburg impedance" is therefore modeled as a resistor and a capacitor in series.

Chemical reactions may also limit the rate of current flow. The equivalent electrical impedance of chemical reactions was investigated by Gerischer (23) who showed that it, too, has real and imaginary components and could also be modeled as a resistor in series with a capacitor. The diffusion into or release of metal "ad-atoms" from the ordered metallic lattice of a corroding surface may also impede the flow of electrons. The term "ad atoms" was coined by Lorenz (24) and is the word which describes the final state of the metal ion prior to going into solution. Just as in the case of diffusion and reaction impedance, this "crystallization impedance" has real and imaginary parts and is modeled by Vetter (25) as a capacitor in parallel with a resistor.

Solution Resistance

The final component of the general electronic network model is the resistance of the solution between the reference electrode and the thin interface layer where all other components of the network lie. In electroanalytical studies of redox reactions, one usually adds an innocuous supporting electrolyte to the solution which eliminates or minimizes the contribution of this component to the overall impedance. In corrosion studies, the addition of a highly conductive salt such as KCl may prove anything but innocuous, leading to a significant increase in corrosion rate. Furthermore, if one considers the application of a general network model to a variety of corrosion monitoring scenarios,

there would be great utility in being able to monitor corrosion where the magniture of the solution resistance is quite large such as in the corrosion of reinforcing bars in concrete or in the corrosion of metals in organic solutions with only small concentrations of electrolyte.

Equivalent Circuit Models

The general electronic network model for an electrochemical interface with all the aforementioned equivalent electrical impedances is discussed by Vetter (26) and is depicted in Figure 2.1. Grahame (27) introduced the concept of faradaic impedance to describe the effective impedance of the series-connected upper loop of the general model. Other investigations have sought to simplify the general model by neglecting components whose contribution to the total impedance would be small under particular circumstances. For example, an equivalent circuit diagram which considers only charge transfer and diffusion components of the faradaic impedance was introduced by Randles (28) and is shown in Figure 2.2.

Such simplifying models as those put forth by Randles are justified in view of the fact that the impedance offered by some components of the general electronic network model may be orders of magnitude smaller than others. The simplified models are also less difficult to analyze than the general model. The method of determining which components to neglect may not be very straightforward, however. Vetter (29) discusses the ramifications of this problem at length and describes a variety of experimental methods useful in electrochemical analysis. For example, either diffusion or a chemical reaction may be the cause of a limiting current in a direct current polarization test. One may distinguish

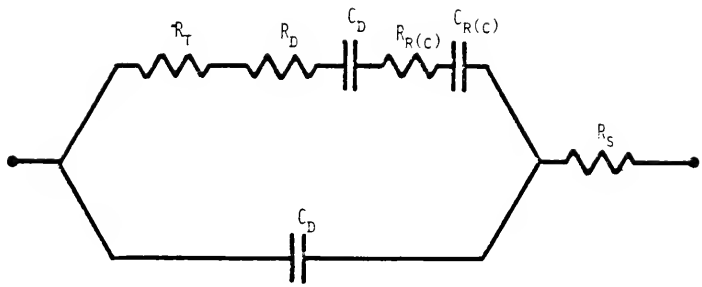


Figure 2.1 General electronic network model for an electrode showing impedances associated with charge transfer (R_t), diffusion (R_d,C_d), chemical reaction (R_r,C_r), crystallization (R_c,C_c) and also considering double layer capacity (C_D) and solution resistance R_s (after Vetter).

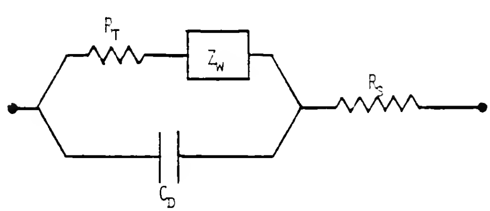


Figure 2.2 Randles circuit model for electrode impedance, W, Warburg impedance associated with diffusion; R_t, charge transfer resistance; R_s, solution resistance; C_D, double layer capacitance.

which of the two provides the dominant resistive component by stirring. Reaction impedances are not affected by stirring while diffusion impedances are decreased due to diffusion-layer thinning.

Alternating current measurements may also be used to distinguish among and to quantify the various impedance components. Since the impedance of a capacitive element is a function of frequency, plots representing the dependence of the faradaic impedance (upper half of the circuit shown in Figure 2.1) on the square root of the reciprocal frequency are sometimes useful in separating the various contributions to the faradaic impedance. Such a plot for an interface exhibiting only charge transfer and diffusion resistance is shown in Figure 2.3. The upper line represents the real part of the impedance and contains only the resistive elements R_t and R_d while the lower curve is the imaginary part of the impedance and contains only the capacitive part of the diffusion impedance. Of course, interfaces exhibiting other contributions to the faradaic impedance face more difficult interpretation with this method. Furthermore, experimental techniques of separating the faradaic component from the double-layer and solution-resistance components do not usually work in electrochemical reactions in which the electrode itself is involved, i.e., corrosion.

Electrochemical Reactions in the Presence of Passive Films

In addition to the impedance contributions of the electrochemical interface, the character of corrosion product films is of extreme importance to the rate and type of corrosion for many metals. Indeed, the presence of a passive film may protect an otherwise active metal from electrochemical deterioration.

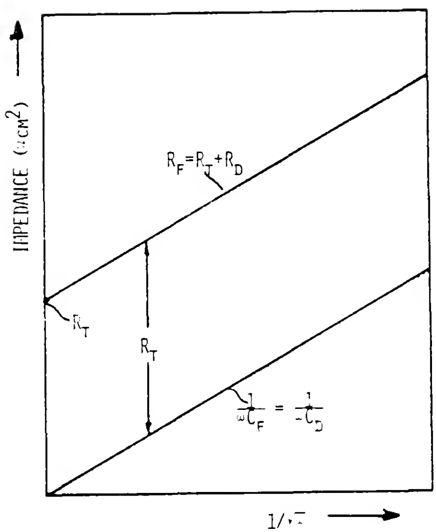


Figure 2.3 Dependence of the components of the faradaic impedance on $1/\sqrt{\omega}$ for rate-control determined by diffusion and charge-transfer only; R_f, faradaic resistance; R_t, charge transfer resistance; R_d, diffusion resistance; $1/\omega C_f = 1/\omega C_d$, capacitive reactance due to diffusion.

Impedance data for solid films suggest that a series or parallel combination of a resistor and capacitor can be used as a model for the film. Pryor (30), Beck et al. (31, 32), Heine et al. (33-35) and Richardson et al. (36, 37) have conducted numerous studies of the properties of both air-formed and anodic oxide films on aluminum using AC impedance techniques. In the attempt to isolate the impedance of the film from the impedance contributions of the electrochemical interface, Richardson et al. (37) proposed the equivalent circuit model as shown in Figure 2.4. Haruyama and Tsuru (38) also considered this so-called dielectric film model and have contrasted it with charge transfer and adsorbed oxygen models in predicting the impedance characteristics of passive iron.

Electrochemical Corrosion Monitoring

In the absence of a general predictive theory for the rate of metallic deterioration under the wide variety of possible exposure conditions, engineers rely on empirical corrosion rate data obtained from systems closely resembling the one of interest. Such data collection methods, using direct analytical methods such as weight loss or spectroscopic solution analysis, are time-consuming and, even when carried out carefully, may not accurately represent the full range of environmental conditions present in a real system. The direct analytical methods are also limited to metallic systems which do not form adherent layers of corrosion product.

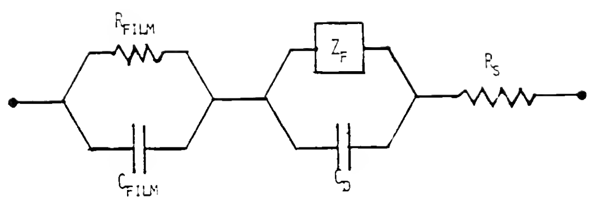


Figure 2.4 Richardson, Wood, Breen model of an electrochemical interface with a passive film on the surface; C_F capacitance of film; R_f, film resistance; C_D, double layer capacitance; Z_f, faradaic impedance; R_S, solution resistance.

The electrochemical mechanism of metallic deterioration in aqueous environments has led to the application of electroanalytical methods to the study of corrosion and to the in situ monitoring of corrosion in real systems. Electrochemical methods can determine corrosion rates much more quickly than the direct methods and are reasonably accurate. However, most electrochemical methods suffer from the disadvantage that they must perturb the corroding system with an externally applied DC voltage, a fact which inevitably changes the local surface properties and perhaps the local corrosion rate from that of the surroundings. Recognizing the potential adverse consequences of this perturbation, one seeks a method which obtains information about the corrosion process as quickly as possible with the smallest possible perturbations.

Two generic types of electrochemical methods have been applied to the corroding interface: the widely-used direct current polarization methods and the more recent AC impedance techniques.

DC Methods

The two most popular DC methods are Tafel line extrapolation and polarization resistance measurements.

Tafel line extrapolation. The method of Tafel line extrapolation is based on the theories of Wagner and Traud (39) and employs large polarization amplitudes. By extrapolating the large amplitude cathodic and anodic polarization curves toward the corrosion potential, one obtains I_{corr} and E_{corr} from the point of intersection. See Figure 2.5. This method is widely used as a laboratory analysis technique, but because of the large polarization of the corroding electrode, it can cause irreversible changes during the measurement process. This fact renders it of only limited value for corrosion monitoring purposes, in

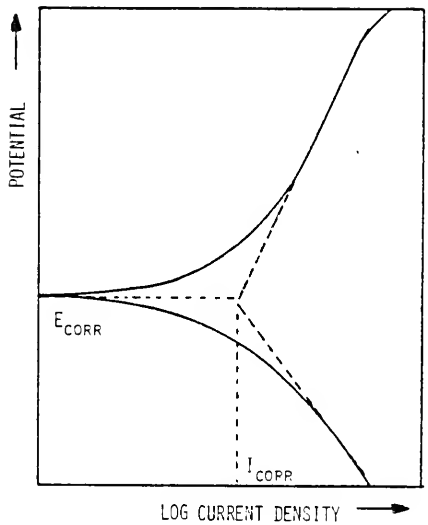


Figure 2.5 Determination of corrosion potential and corrosion current from Tafel line extrapolation.

and of itself, although the determination of Tafel line slopes is required for corrosion rate computation via the polarization resistance method. (See discussion of polarization resistance below.)

Lorenz and Mansfeld (11) point out that the corrosion rates predicted by Tafel line extrapolation for uniform metal corrosion in acid media are in good agreement with weight loss measurements. However, in systems where corrosion product layers form on the surface, predictions of corrosion rate via the Tafel line method may be very inaccurate. This is not surprising since the imposition of a DC signal may change the characteristics of the corrosion product film. Furthermore, breakdown of the film tends to occur locally rather than uniformly over the surface. Thus, the "average" corrosion current does not accurately reflect the destruction of the component. Similar predictive errors have been observed in the presence of inhibitors (11).

Polarization resistance. Measurement of polarization resistance is a DC method more suitable for use in corrosion monitoring. Polarization resistance, R_p, is defined as the tangent to a polarization curve at the corrosion potential. See Figure 2.6. The relationship between the polarization resistance, R_p, and corrosion current, i_{corr}, was developed by Stern and Geary (40, 41) and Stern (42). They showed that for a simple charge transfer controlled system,

$$i_{corr} = \frac{\beta_a \ \beta_c}{2.303\ (\beta_a + \beta_c)} \ \frac{1}{R_p} \qquad (2.3)$$

where i_{corr} is the corrosion current and R_p is the polarization resistance. Since β_a and β_c are the anodic and cathodic Tafel constants,

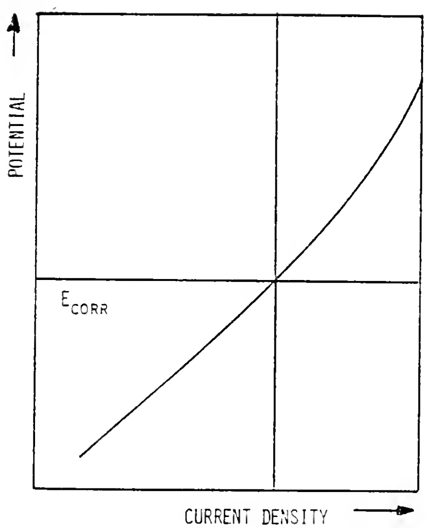


Figure 2.6 Determination of Polarization Resistance

Equation (2.1) can be written

$$i_{corr} = \frac{B}{R_p} \tag{2.4}$$

where

$$B = \frac{\beta_a \ \beta_c}{2.303(\beta_a + \beta_c)}$$

Thus, the determination of anodic and cathodic Tafel constants and the slope of the polarization curve yield predictions for i_{corr} via Equation (2.4).

Polarization resistance measurements are generally made at applied potentials within 30 mv of the corrosion potential in an attempt to confine polarization to the linear region. Although R_p can be measured with AC or DC perturbations, the majority are made using DC steady-state techniques. The DC steady-state techniques can be very time consuming where corrosion rates are very low, a disadvantage for a monitoring technique. The value of polarization resistance measured by such a technique also contains a contribution from solution ohmic resistance. When this so-called "uncompensated resistance" (R_s) is large, the error due to its inclusion is considerable and if not accounted for leads to underestimation of corrosion rate.

Other sources of error in polarization resistance measurements are discussed by Lorenz and Mansfeld (11) and reviewed extensively by Callow et al. (43). Among the many factors mentioned are failure to achieve steady-state during polarization; time-dependence of the corrosion phenomenon, particularly during early stages; localized corrosion processes such as pitting or crevice corrosion; hydrogen absorption and adsorption; adsorption of reaction intermediates; and inhibitor redox processes.

AC Methods

While the various DC methods all attempt to characterize the corroding interface in terms of a single resistance value, AC methods offer, in principle, the possibility of separating solution and faradaic resistance components and permit simultaneous quantification of the capacitive component of the complex impedance. Because they are capable of quantifying both resistive and capacitive impedance components, AC methods offer much more latitude in establishing the efficacy of the theoretical models described previously. Numerous investigations have capitalized on this capability and have used AC methods to measure polarization impedance, the impedance of anodic films and faradaic impedance of redox reactions. Excellent reviews of the development of AC electrochemical methods are given by Grahame (27), Sluyters-Rehbach and Sluyters (44), and Smith (45).

Because of the capacitive components in the electrochemical interface network, polarization impedance is frequency-dependent (31). At high frequencies, the capacitive reactance due to the double layer is low and thus determines the total polarization impedance. Conversely, at low frequencies the capacitive reactance of the double layer approaches infinity and the polarization impedance is equal to the polarization resistance R_p. See Figure 2.7. This realization led Epelboin et al. (1) to define the polarization resistance, R_p, as the limit of the faradaic impedance at zero frequency.

Epelboin and coworkers (1, 2) also suggest that a more reliable correlation with corrosion rates is obtained by using a quantity called the "charge transfer resistance" (R_t), the limit of the faradaic

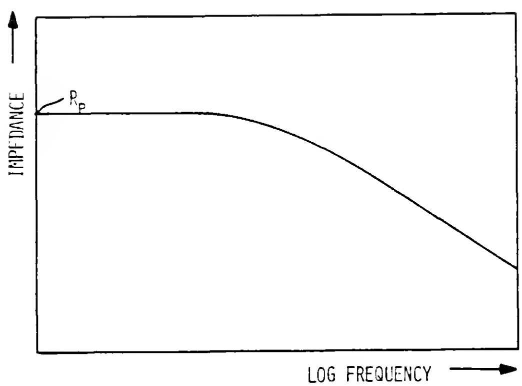


Figure 2.7 Frequency dependence on an electrochemical interface impedance. R_p is the limit of faradaic impedance at zero frequency.

impedance at infinite frequency. By performing measurements at high frequencies, it was reasoned, variations in the surface coverage of adsorbates would be precluded since diffusion could not keep up with the changes in polarity. In a study of the inhibition of iron corrosion by propargyl alcohol in acid solutions, R_t successfully predicted corrosion rates where measurement of R_p failed.

Lorenz and Mansfeld (11) dispute the general usefulness of this approach, however, citing its dependence on the assumption that the double-layer capacitance can be totally separated from other capacitive contributions of the electrochemical interface. The treatment of the double layer as a capacitor in parallel with the polarization resistance leads to the prediction of a single semicircular loop in the complex plane plot of the system impedance. As shown in Figure 2.8, the iron-propargyl alcohol systems investigated by Epelboin et al. deviate substantially from this predicted behavior. The presence of inductive loops in these and other iron systems is a particularly puzzling phenomenon from the standpoint of correlating with any interface model. One basic conclusion of Lorenz and Mansfeld's critique seems particularly apropos: a knowledge of system-specific corrosion behavior is required before any electrochemical measurement methods can be used reliably to predict corrosion rate.

Measuring electrode impedance. The impedance of an electronic network or of an equivalent circuit which simulates a corroding electrode is a complex function of frequency possessing both magnitude and phase information. Having the units of resistance, it may be found by taking the ratio of the complex current flowing through the circuit to the complex voltage drop across the network.

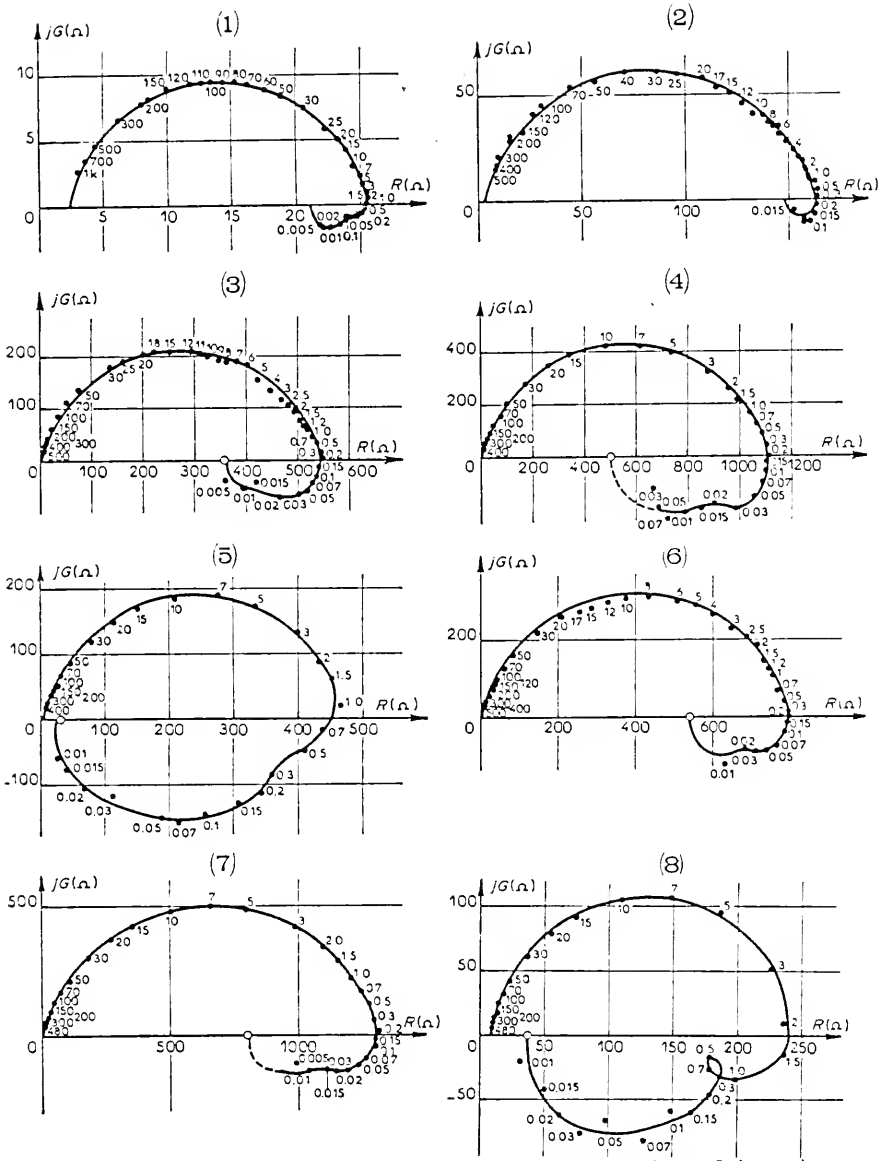


Figure 2.8 Impedance diagrams for spontaneous corrosion of iron in aerated H_2SO_4; (1) 1M H_2SO_4; (2) 0.5 M H_2SO_4; (3) 0.5 M H_2SO_4 + 0.1 x 10^{-3} M propargylic alcohol; (4) 0.5 M H_2SO_4 + 0.2 x 10^{-3} M propargylic alcohol; (5) 0.5 M H_2SO_4 + 0.5 x 10^{-3} M propargylic alcohol; (6) 0.5 M H_2SO_4 + 2 x 10^{-3} M propargylic alcohol; (7) 0.5 M H_2SO_4 + 5 x 10^{-3} M propargylic alcohol; (8) 0.5 M H_2SO_4 + 20 x 10^{-3} M propargylic alcohol. See Reference (1).

This can be done by a number of methods. A simple though tedious approach is to compare the input voltage perturbation with the output current response in x and y channels of an oscilloscope. The resulting Lissajous figure can be used to determine the impedance modulus and phase shift for a single frequency. The method is time-consuming since it must be repeated at each frequency and is not very practical for low frequencies. Another technique compares input and output signals at a single frequency and yields direct reading of modulus and phase shift or, in some cases, real and imaginary components. The tedium of multiple sequential frequency measurements is alleviated somewhat by the availability of programmable equipment, e.g., Solartron. The only method capable of simultaneously comparing the perturbation and response of multiple frequency signals employs equipment which computes the Fourier transform.

Linear System Theory

Assumptions

When the impedance is evaluated with a digital signal analyzer, the interface is treated as a "black box" with a single input and output terminal. The potential drop across the interface is treated as the output function of the system and is compared with the current flow through the interface which is treated as the input function of the system. The algebraic ratio of these two signals expressed as functions of frequency is the frequency response or sometimes called the transfer function of the system. Before one attempts to determine the impedance of a corroding interface by means of digital signal analysis and to use it to characterize the interface, it is appropriate

to consider the inherent assumptions one makes in this process. These assumptions are relaxedness, linearity, and time-invariance (46).

Relaxedness. When any physical system is treated as a black box and one is attempting to abstract key properties of the system from its response to some excitation, one must be certain that the system is initially relaxed, i.e., that the system is not still responding to some previously applied signal at the instant of test signal application. For such a relaxed system, the response y(t) may be related to the input excitation u(t) by the following relationship:

$$y(t) = h \cdot u(t) \tag{2.5}$$

where h is a function that uniquely specifies the output y(t) in terms of the input u(t).

Relaxedness is usually a justifiable assumption when evaluating electrochemical interfaces. From the author's own experience, the relaxation time of an electrochemical interface from a pulse is typically less than one second. Thus, if the interface has not been stimulated for ten seconds or more, the response of the system may be reasonably assumed to have resulted only from the excitation u(t). Of course, all electrochemical interfaces exhibit continuous random fluctuations in potential and current usually known as electrochemical noise (14, 47). These fluctuations typically have an amplitude on the order of μV, and the system obviously exhibits continuous response to these excitations. However, if one makes the amplitude of the excitation signal large enough, the ratio of input/output signals to system noise is sufficient to obscure the frequency response due to system noise.

Linearity. A relaxed system is considered linear if two mathematical conditions are satisfied (46): (1) the output due to a combination of inputs equals the sum of the outputs due to each input applied individually, i.e.,

$$h \cdot \{\sum_i u_i\ (t)\} = \sum_i h \cdot u_i(t) \tag{2.6}$$

and (2) the output due to an individual input multiplied by a scalar equals the output multiplied by that scalar, i.e.,

$$h \cdot \{\alpha\ u_i(t)\ \} = \alpha h \cdot u_i(t) \tag{2.7}$$

A linear electrochemical system exhibits a linear relationship between current and potential for all values of frequency. As shown previously by Equation (2.2), the charge transfer behavior is approximately linear if the polarization voltage is kept close enough to equilibrium. This implies that Equation (2.7) is valid only for values of α below some upper bound. The limit of input signal amplitude can be determined in practice by the limit of linearity in a DC polarization experiment. In general, input signal amplitudes less than about 30 mv from equilibrium satisfy this requirement.

Time-invariance. A relaxed linear system is time-invariant if the characteristics of the system do not change with time. This condition is not rigorously valid for the electrochemical interface of a corroding electrode, particularly when a passive film is present. In such cases, the gradual growth of the film will have a definite effect on the values of the network parameters. To get around this limitation, one must assume that the corroding electrode exhibits quasi-steady state behavior, i.e., does not change significantly during the period

of measurement. Minimizing the period of measurement, besides enhancing characterization speed, also serves to assure that this condition is satisfied.

Conclusions. The requirements of relaxedness, linearity and time-invariance impose some important constraints on the way electrode impedances may be measured using digital signal analysis. In order to neglect the effect of random fluctuations, the excitation signal must have a significantly larger amplitude than the system noise. To satisfy linearity requirements, on the other hand, the input signal should be as small as possible, usually less than about 30 mv from equilibrium. To assume that time-invariance of the system is adequately approximated, the period of measurement should be as short as possible.

Domain Transformations

The transformation of time domain signals into the frequency domain for the purpose of computing electrode impedance is outlined by Pilla (48). Pilla illustrates how this transformation might be accomplished in general by the Laplace transformation:

$$F(s) = \int_{0}^{\infty} f(t) \exp(-s)dt \tag{2.8}$$

where s is the Laplace transform variable (49). The quantity s is a complex number given by $s = \sigma + j\omega$ in which σ is the real and $j\omega$ the imaginary part. Due to the properties of F(s), it is possible to integrate along either or both the real and imaginary axes is the complex frequency plane which defines F(s). In the imaginary axis transformation ($s = j\omega$), Equation (2.8) may be written:

$$F(j\omega) = \int_{0}^{\infty} f(t)\ \exp\ (-j\omega)dt \tag{2.9}$$

Equation (2.9) is the well-known single-sided Fourier Transform (48). For relaxed, linear, time-invariant systems, the frequency response function may replace the transfer function with no loss of useful information (49).

Although some investigations continued to be made into the utility of Laplace transformations (50, 51), the development of the "fast Fourier Transform" algorithm (FFT) (52, 53) made it convenient and practical to effect the imaginary axis transformation in real time. Creason and coworkers (5-8) capitalized on this development, demonstrating that with an on-line minicomputer, it is possible to acquire and transform time domain signals into the frequency domain in less than 3 seconds (6).

Utilizing on-line computation of the FFT, it is possible to determine the admittance of an electrochemical cell by the expression

$$A(\omega) = I(\omega)/E(\omega) \tag{2.10}$$

where $A(\omega)$ is the cell admittance, $I(\omega)$ is the cell current, and $E(\omega)$ is the potential across the electrical double layer. For the sake of convenience of expression, Creason and coworkers (6) chose the alternative form

$$A(\omega) = I(\omega)\cdot E^*(\omega)/E(\omega)E^*(\omega) \tag{2.11}$$

where $E^*(\omega)$ is the complex conjugate of $E(\omega)$. In this formulation, the admittance is expressed as the cross power spectrum divided by the auto power spectrum, a form which produces phase information in the numerator only. See Appendix A.

Although, in principle, this formulation is applicable to any generalized test signal, Creason and coworkers anticipated that some signal waveforms might be more "efficient" than others (6-8). In this context, "efficiency" refers to the amount of data dispersion present after a certain number of replicate measurements. With this in mind, they undertook a detailed empirical study of measurement efficiency associated with Fourier transform faradaic admittance measurements (8). Four waveform classes were used: (1) complex periodic signals, waveforms composed of discrete coherently-related sinusoidal components; (2) almost periodic signals, waveforms composed of discrete non-coherently related sinusoidal components; (3) aperiodic transients, signals with continuous well-defined smoothly-varying phase and amplitude spectra; and (4) stochastical signals, signals with continuous spectra which have smooth distribution after long times.

The efficiency of the various waveforms was evaluated using the redox couple $Cr(CN)_6^{-3}/Cr(CN)_6^{-4}$ in 1 M KCN at 25°C on a dropping mercury working electrode. As is customary for AC polarographic experiments, the faradaic admittance data is portrayed in plots of magnitude vs $\omega^{\frac{1}{2}}$ and cot ω vs $\omega^{\frac{1}{2}}$. See Figure 2.9. Such plots are linear or nearly so for the redox couple under consideration, and it was thus possible to quantitatively assess measurement precision based on the relative standard deviations of the intercept and slope as determined by the linear regression technique.

The results of this comparison for 64 replicates on random noise, pseudo-random noise, pulse and multicomponent sinusoidal arrays of varying amplitude is quite dramatic. In general, the data precision

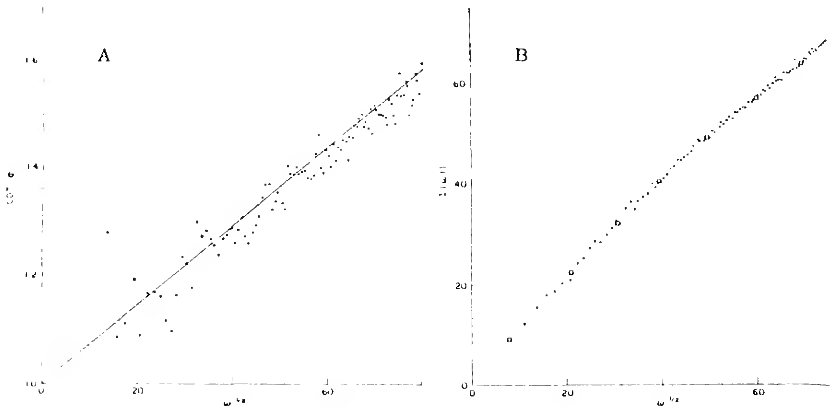


Figure 2.9 Frequency response of $Cr(CN)_6^{3-}/Cr(CN)_6^{4-}$ in 1 MKCN. Plots depict cotangent of phase angle and fundamental harmonic peak current as function of frequency when stimulated by bandwidth limited white noise. Data represent average of 16 replicate runs. See Reference (6).

improved with decreasing signal amplitude presumably in response to decreases in faradaic non-linearity. Overall, the best precision was achieved with a phase-varying 15 component odd harmonic array with a standard deviation in the intercept of 0.08%. Under a similar set of measurement conditions, a negative pulse produced an intercept standard deviation of 6.33%. The random noise and pseudo-random noise signals fell between these extremes with standard deviations of 1.39% and 0.56%, respectively.

Based on these early data, Schwall et al. (12) developed a high speed synchronous data generation and sampler system for which the acronym SYDAGES was coined. SYDAGES functions as a programmable signal generator combined with two data acquisition channels with the capability of handling signals up to 500 kHz.

Smith (10) has predicted that Fourier transform data processing on electroanalytical measurements will exceed its influence in the field of spectroscopy. Despite this predictions, relatively few investigators have conducted studies utilizing this feature. Blanc et al. (4) demonstrated how impedance measurements could be made on an iron-sulfuric acid system using a so-called correlator, a device which determined the Fourier transform. DeLevie and coworkers (13) have applied the procedure in the study of ion-conducting ultrathin membranes. Smyrl and Pohlman (54) demonstrated that corrosion parameters can be determined by this method although in their system, the Fourier Transform was performed batchwise on a CDC 6600 system.

The apparent reticence of the electrochemical community to embrace this technique may be explained in part by the fact that

electronic system noise becomes a problem when one tries to perturb an electrochemical interface with small multiple frequency signals. Unlike the single frequency "lock-in amplifier" method, there is no way to distinguish between electronic system noise and system response in developing the Fourier spectrum. Fortunately, investigators of electrochemical noise (3, 14, 47) have also been concerned with this problem. Recent work by Schideler and Bertocci (47) has resulted in the development of low-noise potentiostat capable of suppressing electronic noise to the order of 2.5×10^{-8}v/Hz. They (47) have used this low-noise potentiostat to measure electrode impedance using both superimposed and electrochemical noise signals.

Analysis of AC Impedance Data

During the evolution of AC frequency response techniques, a number of methods of portraying and analyzing data have been proposed and used. Sluyters-Rehbach and Sluyters (44) provide a review of the possibilities as they might apply to electrochemical analysis. One of the most common portrayals in the AC literature is the complex plane plot, that of the imaginary component of the electrode impedance plotted against the real part.

For a Randles equivalent circuit (see Figure 2.2), the complex plane plot can provide quantitative information about the various components of the circuit. Such an equivalent circuit will produce a complex plane plot of the form shown in Figure 2.10. As can be seen in Figure 2.10, the plot may be separated into two regions, one exhibiting semicircular character and the other linear behavior. This observation is consistent with the analytical determination of

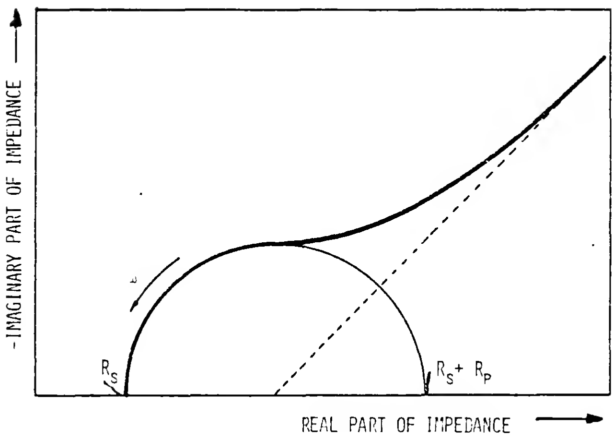


Figure 2.10 Analytical prediction of impedance behavior of Randles equivalent circuit.

Sluyters-Rehbach and Sluyters (44) who showed that the real and imaginary components of the Warburg impedance can be determined in the low frequency linear region while the semicircular high frequency region yields quantitative data on the values of the remaining components.

If experimental acquisitions of impedance data yields no such linear low frequency region, which is often the case for corroding electrodes, the circuit can be modeled as the three-element networks shown in Figure 2.11. The impedance of the network shown in Figure 2.11 may be written:

$$\hat{Z} = R_s + \frac{R_p}{1 + j\omega\, R_p C_d} \tag{2.12}$$

The real part of the impedance is given by

$$Z_R = R_s + \frac{R_p}{1 + \omega^2 C_D{}^2 R_p{}^2} \tag{2.13}$$

and the imaginary part is given by

$$Z_I = -\frac{\omega\, C_D R_p{}^2}{1 + \omega^2 C_D{}^2 R_p{}^2} \tag{2.14}$$

By appropriate rearrangement and combination of Equations (2.13) and (2.14), one finds

$$\omega = -\frac{Z_I}{(Z_R - R_s) R_p C_D} \tag{2.15}$$

which upon substitution into Equation (2.13) and the appropriate rearrangement yields

$$\{Z_r - R_s - \frac{R_p}{2}\}^2 + Z_I{}^2 = \frac{R_p}{2}^2 \tag{2.16}$$

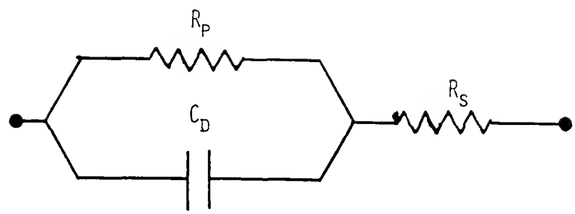


Figure 2.11 Simplication of Randles equivalent circuit; valid where diffusion is not rate-limiting.

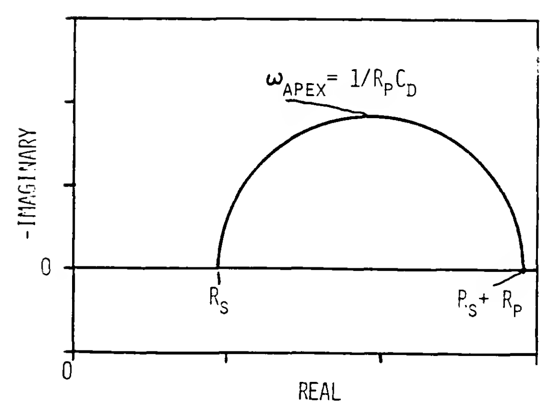


Figure 2.12 Complex plane evaluation of the three-element network shown in Figure 2.11.

This is the equation of a circle in the complex plane with its center on the Z_R axis at $Z_R \doteq R_s + \frac{R_p}{2}$ and radius $R_p/2$. Intersections of the circle with the Z_R axis occur at $Z_r = R_s$ for $\omega = \infty$ and $Z_R = R_s + R_p$ for $\omega = 0$. Only cases where ω, C_D, R_p and R_s are positive have physical significance; so one plots the circle only in the fourth quadrant of the complex plane. In the literature, this plot is usually rendered as $-Z_I$ vs Z_R as shown in Figure 2.12. It can also be shown that the frequency at $Z_R = R_s + R_p/2$, i.e., at the apex of the semicircle, can be given by

$$\omega_{apex} = \frac{1}{R_p C_D} \tag{2.17}$$

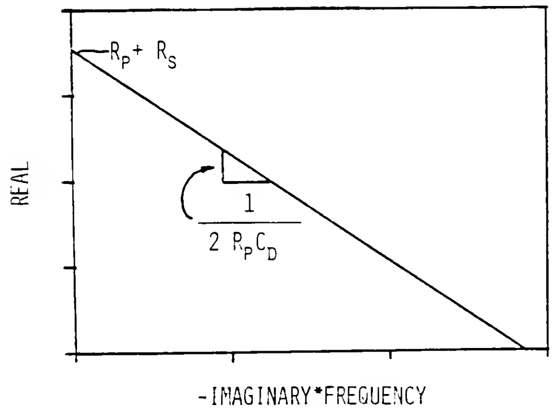
Given data over a sufficient range of frequency, therefore, one can determine values for R_s, R_p and C_D from a complex plane plot of the impedance.

Other methods of graphical analysis have been suggested in the literature (54). Linear plots are useful because it is difficult to curve-fit a semicircle when there is data scatter. By combining Equations (2.13) and (2.14), two linear equations are derived:

$$Z_R \doteq R_s + R_p - R_p C_D \, \omega \, Z_I \tag{2.18}$$

$$Z_R = R_s + \frac{Z_I}{\omega \, R_p C_D} \tag{2.19}$$

Plotting Z_R vs Z_I*f yields a straight line with a slope of $- 2\pi \, R_p C_D$ and an intercept of $R_s + R_p$. A plot of Z_R vs Z_I/f provides an intercept of R_s. See Figure 2.14.



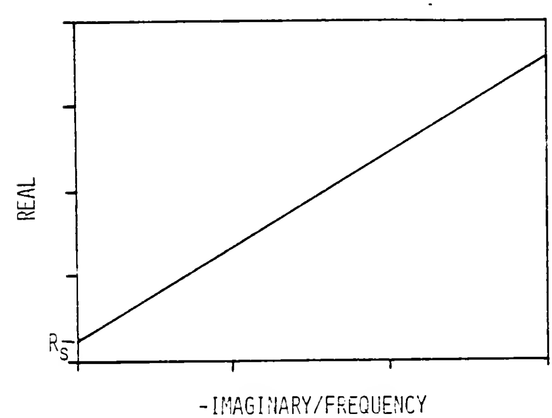


Figure 2.13 Pohlman-Smyrl technique of analyzing the impedance of the three-element network given in Figure 2.11.

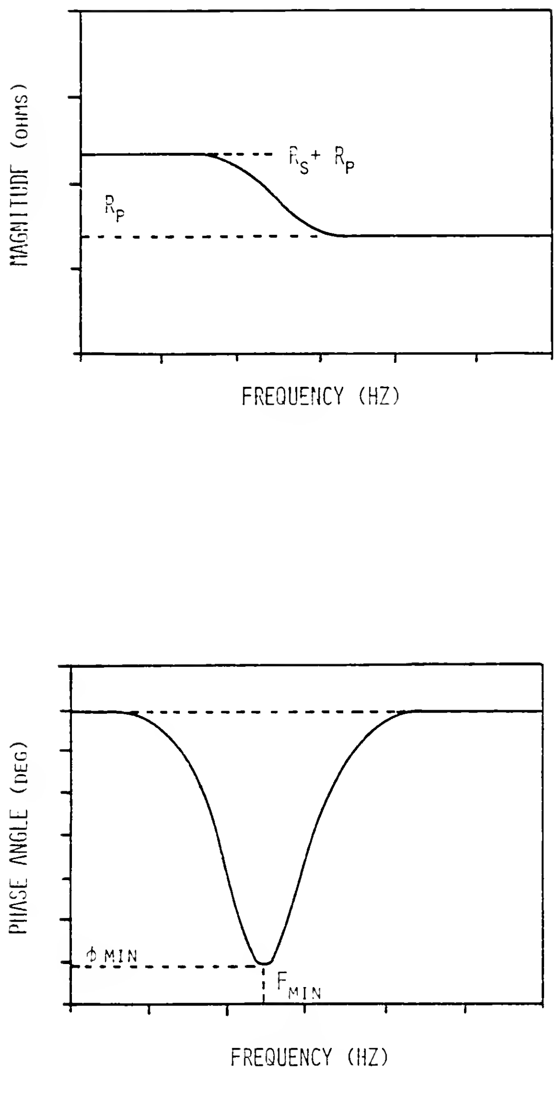


Figure 2.14 Bode plot of frequency response for the three-element network shown in Figure 2.11. ϕ_{min} and f_{min} are complicated functions of C_D and R_p.

A third common method of data portrayal is done with plots of log $|\hat{Z}|$, and phase angle, ϕ, vs log of frequency, f, commonly known as Bode plots. The values of R_p and R_s may be obtained from the former plot as shown in Figure 2.14 and in combination with the latter plot yields C_D. The Bode plots have several advantages over the other graphical portrayals: (1) since the log frequency scale is used, data from lower frequencies are not obscured; (2) the network component values may be computed using higher frequency data than with the other methods. This is fortunate since the low frequency data require much more time to gather and are more susceptible to significant dispersion.

Recapitulation: State-of-the-Art

The characterization of electrochemical interfaces has been attempted with a variety of methods. DC techniques result in the depiction of the interface as a single polarization resistance term, which includes the resistance of the electrolyte solution. AC techniques offer the possibility of separating solution resistance from faradaic resistance components while quantifying capacitive components at the same time. In Figure 2.15, schematic diagrams of electrochemical cells contrast the interface model using DC polarization with one possible model using an AC technique. AC impedance may be measured by a variety of methods, the majority of which require the sequential application of a series of single frequency sinusoidal signals.

The tedium and time-consumption of such processes is eliminated in principle by fast Fourier transform technology. Using the FFT algorithm in an on-line minicomputer, one can simultaneously investigate a continuum of frequency by converting time domain input perturbation

(A)

(B)

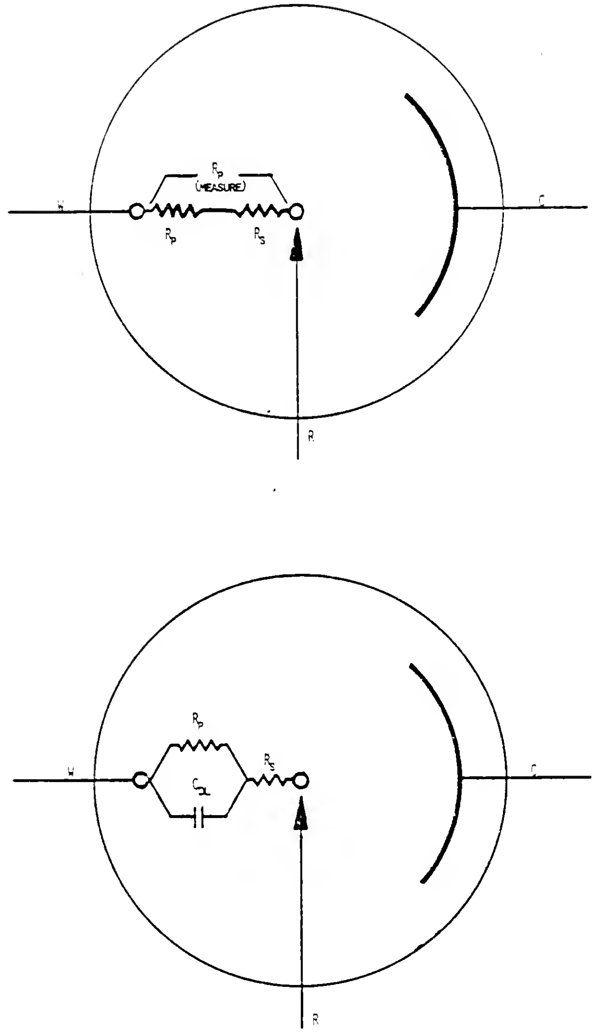


Figure 2.15 Schematic diagrams of electrochemical cells. The interface model is represented as shown in (A) by the DC polarization technique. The three-element model including double layer capacitance as shown (B) or other more complicated models may be depicted using AC techniques. R_p, polarization resistance; R_S, solution resistance; C_{DL}, double layer capacitance; C, counter electrode; R, reference electrode, W, working electrode.

and output response signals into the frequency domain and computing the electrode impedance. Graphical methods applied to the resultant complex plane plots result in the determination of equivalent circuit parameters.

The primary difficulties encountered with FFT electrode impedance measurements have been associated with selection of the "most efficient" signal type and signal amplitude. In general, pseudo-random noise effects less data dispersion than white noise, while transient signals such as pulse, ramp, and step tend to be the least efficient. With regard to signal amplitude, one must select a signal large enough to negate the effects of electronic equipment and electrochemical system noise while not introducing faradaic nonlinearities with too large a signal amplitude. The development of low-noise potentiostats should be helpful in permitting this to be done.

CHAPTER III
SYSTEM DEVELOPMENT

System Components

The assemblage of equipment referred to here as the "AC system" was built around a Hewlett-Packard (HP) 5420 digital signal analyzer. The analyzer continuously monitors and digitizes time-varying analog signals corresponding to the perturbation and response of the system under investigation. Having gathered an ensemble of digitized data representing the time domain, the analyzer performs a transformation to the frequency domain via the fast Fourier transform algorithm (FFT). A variety of algebraic manipulations may then be performed on the resultant arrays to yield both time and frequency domain functional relationships which describe the frequency response of the system. (See Appendix A for a more thorough discussion of the capabilities of a digital signal analyzer.)

The signal analyzer capability of primary interest in this study is the rendition of the transfer function. In its general definition, the transfer function of a system is a frequency domain correspondence between perturbation and response signals. Since it is a complex function, it provides both magnitude and phase information at each frequency. In this study, the pertubation, E(t), is a bandwidth limited white noise (BLWN) voltage signal applied between the reference and working electrode of a three-electrode electrochemical cell. The response, I(t), is a time-varying voltage signal directly proportional

to the current passing between the counter and working electrodes. The transfer function, H(f), is therefore a measure of the corroding electrode admittance, Y(f).

$$H(f) = Y(f) = \frac{\text{output}}{\text{input}} = \frac{I(f)}{E(f)} \tag{3.1}$$

By mathematically inverting the ratio of response to perturbation or by reversing the voltage/current leads to the analyzer, one obtains the electrode impedance, Z(f).

$$Z(f) = \frac{1}{Y(f)} = \frac{E(f)}{I(f)} \tag{3.2}$$

A schematic diagram illustrating this analyzer function is shown in Figure 3.1.

As mentioned in the previous chapter, the BLWN signal is not the most "efficient" for electrochemical interface perturbation (8). The BLWN source was used because of its availability, being an integral part of the HP5420. Other signal sources should provide data less susceptible to scatter.

As shown schematically in Figure 3.2, the BLWN signal is injected into the Princeton Applied Research (PAR) 173 potentiostat at the summing junction of the control amplifier. Here it adds to any DC set potential dialed on the potentiometer. The resultant combination of AC and DC voltage components is then maintained between the reference and working electrodes. This combined signal is sensed by the potentiostat electrometer circuit and is fed to one of the two analog-to-digital converter (ADC) channels of the HP5420. The flow of current between the counter electrode and the working electrode in response to the voltage perturbation is sensed by a zero-resistance ammeter which produces a voltage

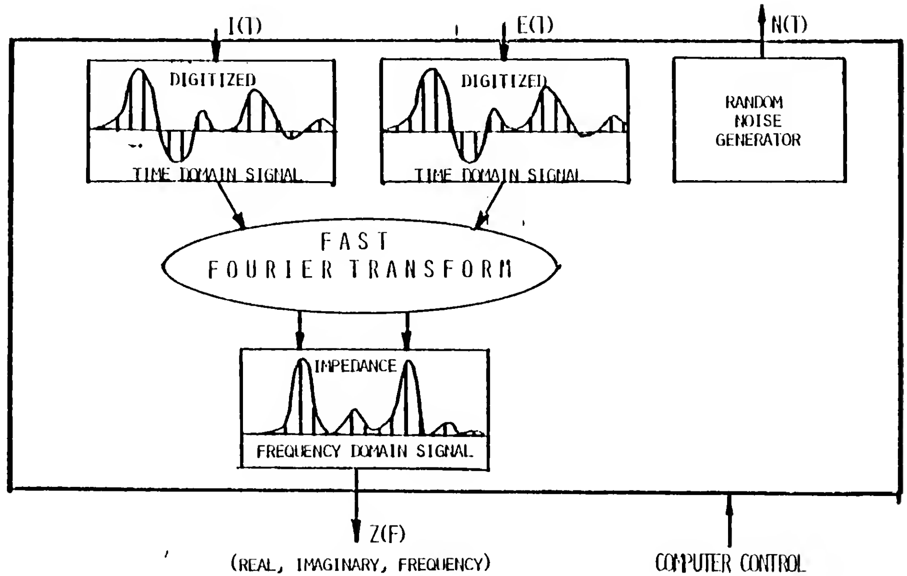


Figure 3.1 Schematic diagram of the HP5420A Digital Signal Analyzer. Under the control of an HP9845 minicomputer, the random noise generator perturbs the electrochemical interface with a voltage perturbation, E(t). The current response of the electrode, I(t), and E(t) are digitized, transformed, and divided to yield impedance Z(f).

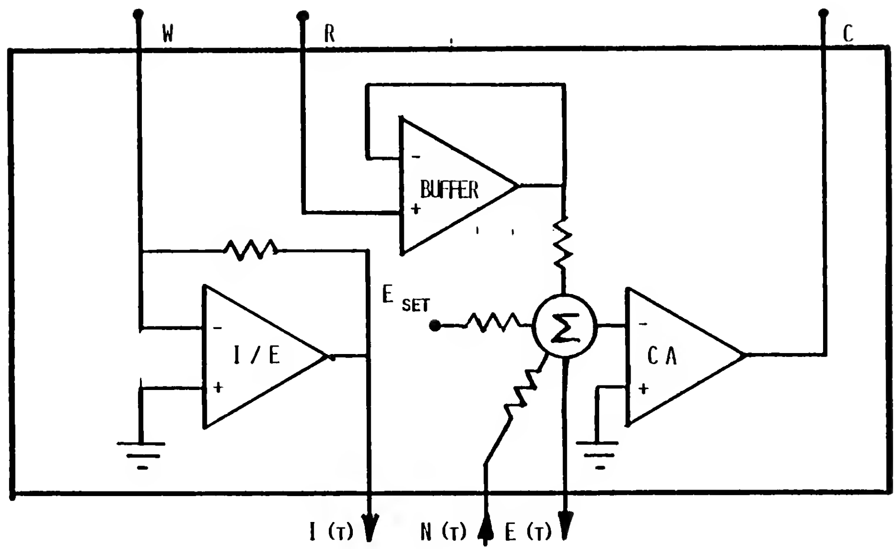


Figure 3.2 Schematic illustration of the PAR 173 potentiostat. The random noise signal, N(t), from the HP 5420A signal analyzer is fed to the summing junction, Σ, of the control amplifier, CA, where it adds to the DC set potential, E_{set}. The resultant voltage signal, E(t), consisting of AC and DC components, is maintained between the reference electrode, R, and the working electrode, W (corroding interface), which is held at virtual ground. The current, I(t), which flows to virtual ground as a result of E(t) is monitored by the current-to-voltage converter, I/E.

proportional to the measured current. This latter voltage is fed to the second ADC channel of the HP5420.

Since the PAR 173 was designed for DC work, there was initially some concern about the capability of the potentiostat to reliably transmit the higher frequency components of the AC signal. To test the frequency response of the potentiostat, a 1 volt p-p sinusoidal signal was applied at the summing junction of the control amplifier. The voltage drop across a resistor connected between the reference and working electrode leads was monitored and compared to the input signal on a dual trace oscilloscope as the signal frequency was increased. There was no perceptible attenuation or phase lag across the resistor until about 30 kHz; above the 25.6 kHz maximum bandwidth range of the HP5420. This procedure demonstrates that the potentiostat is adequate for the range of frequencies which will be analyzed with this technique.

The third major component of the AC system is the 187K byte, HP9845 minicomputer with integral dot-matrix thermal printer and two integral tape drives. To permit even faster retrieval from mass storage, the computer was also equipped with two eight-inch floppy disk drives. The computer carried out a variety of tasks including control of the HP5420, storage of impedance data from individual runs, tying data from sequential runs, mathematical manipulation of data and graphical portrayal of the data on the HP9845 CRT, thermal printer or remote, four-color plotter.

A schematic diagram illustrating the interconnection of the various components is given in Figure 3.3. The HP9845 is connected to the HP5420 via an HP interface bus (HPIB); all other signal-carrying connections being made with coaxial cable.

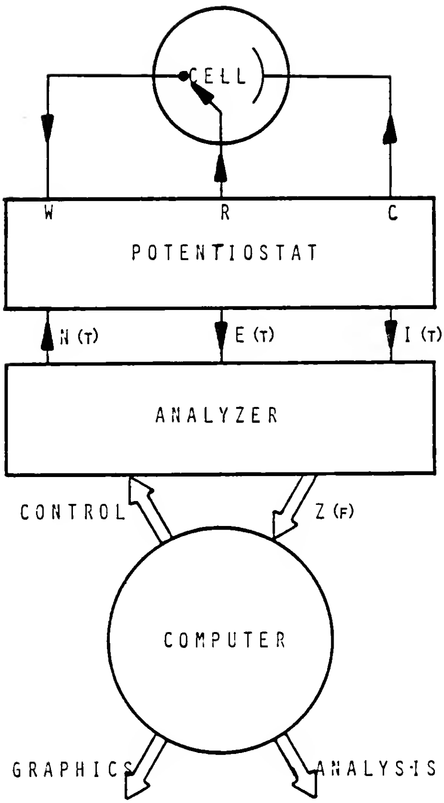


Figure 3.3 Schematic illustrating the interconnection of the electrochemical cell, PAR 173 potentiostat, HP 5420A signal analyzer and HP 9845 computer. The computer controls the operation of the analyzer and receives impedance data from it in the form of real, imaginary, frequency triples of data. Following data manipulation, the computer delivers graphical portrayals of the impedance data.

It would have, of course, been possible to control the set-up and execution of an analysis run by means of manual key strokes on the HP5420 console. However, the versatility of the HP5420 renders the number of keystrokes required to establish a desired set-up state rather large--typically 15-20. By creating command sequences in the software of the HP9845, tedium-induced operator error in the execution of an analysis run can be eliminated. The time required for a set of sequential analysis runs on the same electrode is thus minimized and the runs are reproducible. Furthermore, the HP5420 is incapable by itself of connecting sequentially gathered sets of data, a task necessitated by developmental problems discussed in the next section.

Mathematical manipulation of data includes such tasks as multiplying and dividing the imaginary part of the impedance data by frequency for portrayal in plots of real versus imaginary x frequency and versus imaginary/frequency. The computer also determines the magnitude of the impedance from real and imaginary parts, computes log magnitude versus log frequency plots and scales the data by the value of the resistor across which the current is determined. A linear regression routine may be performed on any selected section of plot yielding the equation of the straight line fit and correlation coefficient. A data-smoothing routine may also be performed on data sets.

Once the desired mathematical manipulations have been performed on the data, they may be portrayed graphically in any of the standard formats described in Chapter II or in any other desired format. Graphical analysis of the standard formats presumes the data fit the

three-component equivalent circuit model described previously, yielding two values of resistance and one of capacitance. One may also generate a theoretical data set for a three-component network with arbitrary values of R_S, R_P and C_D and plot the theoretical set against the real data.

The numerous capabilities of the computer are all controlled from a single main program written in BASIC. The main program is read into the random access memory from tape or floppy disk. Once the user selects the desired computer function from the menu displayed on the CRT, other sections of code required to perform the specified task are read in from floppy disk at the direction of the main program. The main program is interactive with the user thus permitting operation by persons not highly trained in computer technology. As previously mentioned, the user may also create command files of frequently-repeated operations which may be stored or reexecuted. Appendix D contains a thorough description of both the main program and some frequently-used command files.

Developmental Problems

Equipment Integration

At the outset of the project, it was anticipated that equipment integration would be a relatively simple task. During attempts to interface the HP5420 and its integral random noise source to the PAR 173, however, offset voltages were encountered, presumably the result of dissimilar ground loop currents. These offset potentials were troublesome because they changed the DC component of the applied signal. As a consequence, the set potential of the potentiostat was observed to change by as much as 30 mv. In addition, the DC components of the perturbation and response signals were part of the input to the ADC channels while

only the AC components are of interest from the standpoint of analysis.

The choice of AC coupling in the set-up of the HP5420 is supposed to solve the latter difficulty by eliminating the DC component of the signal. However, in practice, the choice of AC coupling did not eliminate the offset potential; in fact, it seemed to exacerbate the problem. Thus, the use of the analyzer's most sensitive 100 mv range was precluded since the combination of the AC signal with DC offset caused the ADC to overflow.

Three grounds were considered in determining the origin of the offset potentials: earth ground, power ground and circuit or virtual ground. Although one normally assumes earth and power ground to be identical, a several hundred millivolt difference was found in our laboratory. Several virtual grounds were also found to be dissimilar. For example, merely connecting the virtual ground on the potentiostat to the virtual ground of the ADC caused the ADC to overflow on its most sensitive 100 mv range. Since the corroding interface is maintained at the virtual ground of the potentiostat, it is clear that ground loop currents and the associated offset potentials are a logical consequence of these disparate ground potentials.

Conventional isolation techniques were not successful in eliminating the voltage offsets or were accompanied by unacceptable side effects. Common point grounding, for example, reduced the offset potentials only slightly. Because of the impedance associated with capacitors and inductors, both capacitor and transformer coupling would have introduced artifact over some range of frequency.

Optical isolation proved to offer a novel yet practical solution to this dilemma. An optical isolation circuit was designed and built. When used between the noise source and input to the potentiostat, offset potentials were eliminated. The circuit design and operation of the optical coupler is described in Appendix B.

Frequency Resolution

A second major area of difficulty in creating the AC impedance system based on digital signal analysis was caused by a limitation of the signal analyzer itself. When operated in the transfer function mode, the HP5420 creates ensenbles of digitized data from the analog signals injected into its two ADC channels. Each ensemble of data is transformed into 256-element array in frequency domain via the FFT algorithm. Dividing the array coming from channel 2 by the array coming from channel 1 yields the transfer function which in the case of these experiments is the electrode impedance. Once impedance data have been collected for a particular range of frequency, they may either be stored on tape as real, imaginary, frequency triples of data, used directly for display in a variety of common graphical formats on the integral CRT of the HP5420, or be transferred to the HP9845 for storage on floppy disk.

One limitation of the analyzer in the study of electrode impedance is a consequence of frequency resolution. Once a range of frequency analysis has been selected, the analyzer divides the range into $N = 256$ equally spaced frequency intervals. The width of a single interval thus defines the uncertainty in frequency, Δf, for that range, and Δf is always 1/256th of the full scale value of frequency. Although at the upper value of frequency in the range the percent error given by

$100 \times \Delta f/f$ is approximately $\pm$ 0.4%, it becomes $\pm$ 4% one decade below the top and $\pm$ 40% two decades below the top. In the limit, the lowest frequency interval contains all frequencies between 0 and 1/256th of the full-scale frequency.

Since the percent error of a frequency measurement is thus inversely proportional to the frequency at which the measurement is made, it is called 1/f error in the technical literature. The upper decade of frequency contains approximately 225 of the 256 frequency intervals which is another illustration of why lower frequencies are resolved so poorly. Figure 3.4 shows how the frequency resolution changes as a function of frequency for various ranges of analysis.

Work with simulated electrodes suggests that three decades of frequency are useful in characterizing electrodes, and for real electrochemical interfaces, at least five decades are needed if the appropriate range of analysis is not known initially. Since only error less than $\pm$ 4% was considered acceptable, it was decided to analyze over consecutive decades of bandwidth to improve the low frequency resolution. The shaded region of Figure 3.4 depicts the error envelope when data from five successively-measured ranges are tied together discarding low resolution overlapping data.

Since it is not possible to either store or display a connected set of data from sequential runs on the HP5420, the HP9845 performs this function on sets of data previously stored on floppy disk. A schematic illustration of this data-tying exercise is shown in Figure 3.5. The sequences of commands necessary to set up the HP5420, make a run for a particular frequency range, store the data for that run and to repeat

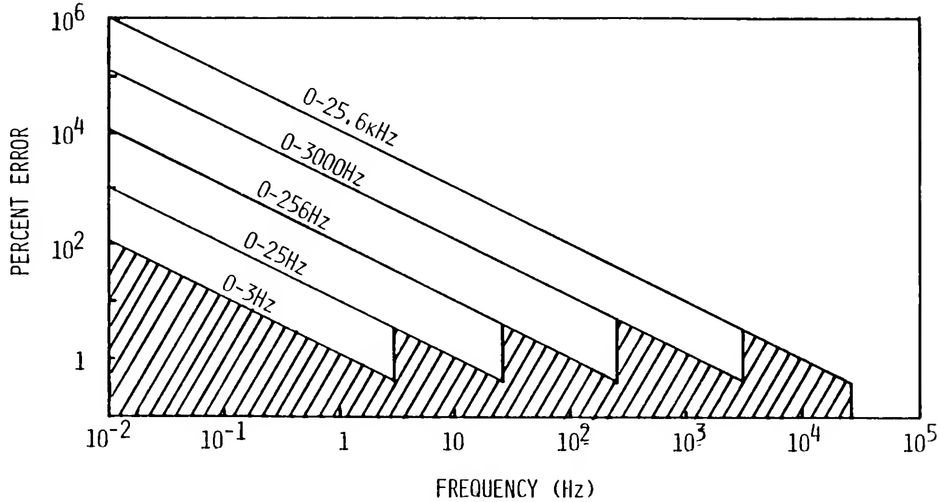


Figure 3.4 Percent error as a function of frequency and frequency range. Shaded region depicts uncertainty in frequency when data from five successfully measured ranges are tied together and low resolution overlapping data points are discarded.

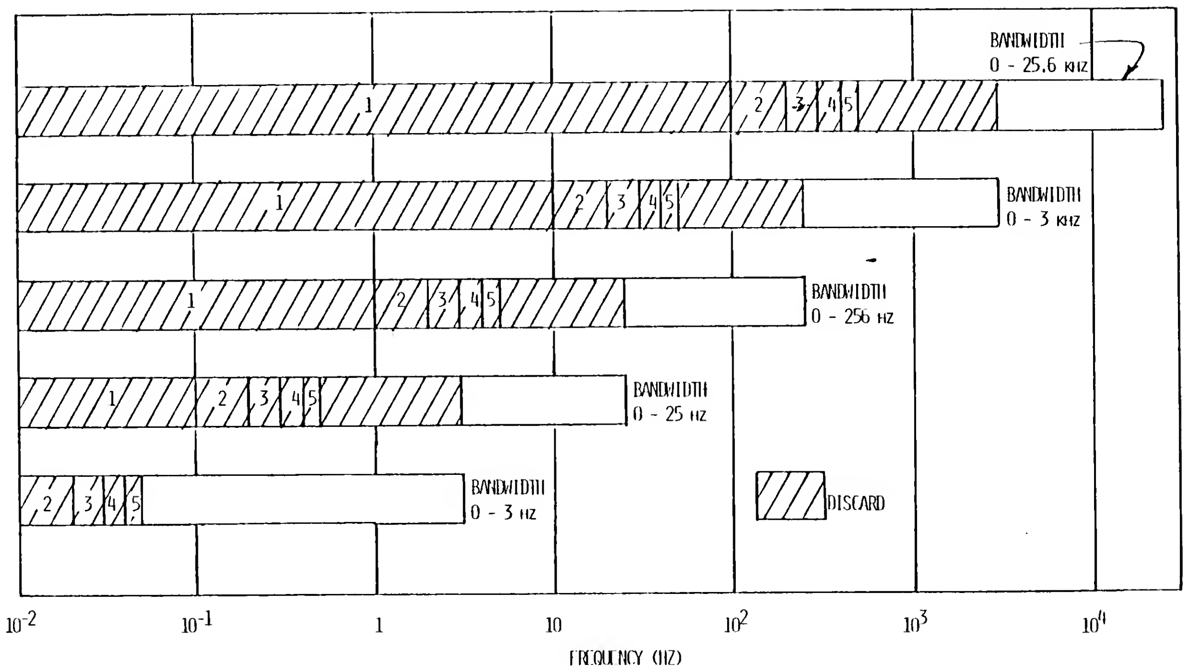


Figure 3.5 Schematic illustration depicting data-tying for five adjacent bandwidth ranges. The shaded regions represent low resolution overlapping data which is discarded. The numbers within the shaded regions show the uncertainty in frequency Δf of the first five channels.

the sequence for four subsequent decades of frequency were written as command files which could be executed by the HP9845. The final step in command files of this type is a data-tying step which consolidates the five individual data sets, discarding low resolution overlapping data. This large data set which consists of greater than 1130 individual points is then stored on floppy disk for subsequent manipulation or graphical portrayal. The command file COMCEL, which controls the HP5420 in the acquisition and tying of data, and other command files are described in more detail in Appendix D.

ADC Resolution--Amplitude Quantization Error

The HP5420 digitizes the analog voltage signals with twelve-bit analog-to-digitizer converters. Since one bit is used to indicate polarity, the converter is able to resolve a full-scale voltage reading into 2^{11} or 2048 discrete voltage levels. Since the most sensitive range of the HP5420 is 100 mv full-scale, the best that the HP5420 can resolve is about 50 μv.

Although investigators of anodic films have used AC signals of 50 mv p-p (23), the general consensus of electrochemists seems to be that one should not perturb an electrochemical interface with a signal greater than 5 mv p-p if nonlinear effects are to be totally avoided. The HP5420 would digitize such a signal by dividing positive and negative voltage excursions of 2.5 mv into 50 discrete voltage levels, respectively. Such resolution is not very good for measurements which must be used in mathematical analysis. Since the FFT is a linear operation, the percent error associated with a time domain signal is carried over into the frequency domain as an equivalent percent error in magnitude.

There are a number of possible ways of dealing with this amplitude quantization error of the HP5420. The first is to accept the resultant penalty in data quality. Gross approximation of voltage values will, of course, result in frequency domain data scatter but will not necessarily destroy the character of graphical portrayals. The second possibility is to use a higher amplitude signal, accepting the risk of non-linear behavior. The third possibility is to amplify the signal before injecting it into the ADC, accepting the deterioration in signal-to-noise ratio brought about by the additional amplifier stage. To satisfy such a requirement, the optical isolators were equipped to provide a gain of 10X, if desired.

All of the above techniques were tried. In the investigations of real electrochemical systems described in the next chapter, it was possible to use signals of higher amplitude without apparent ill-effect. In real systems, however, the electrode impedance changes by orders of magnitude during the formation or dissolution of adherent corrosion product films. Thus, for a given amplitude of voltage perturbation, the current response could vary by orders of magnitude--requiring appropriate ranging of the current-to-voltage converters. To assure that the current monitoring ADC was not subjected to either overflow or underflow conditions, one had to monitor current levels prior to each run and set the range of the current-to-voltage converters accordingly.

Verification of AC System Capabilities

To lend credibility to the assertion that the AC system described herein can be used to reliably characterize an electrochemical interface, experiments were first run on simulated electrodes consisting of

two resistors and one capacitor. Since the double-layer capacitance for an electrochemical interface is widely reported to be on the order of 20 μF/cm^2, an available capacitor with 9.94 μF capacitance was considered representative of the double layer and was used in each of the simulated electrodes. Two values of polarization resistance were used: 996Ω and 49.7Ω, simulating high and low values of polarization resistance, respectively.

To examine the capability of the system to render values for this equivalent polarization resistance, R_p, and the equivalent double-layer capacitance, C_D, when the solution resistance, R_s, varies over a wide range, calculations were first performed to determine what a reasonable upper value for R_s would be in an electrochemical cell with a low conductivity electrolyte. As described in the next chapter, electrochemical experiments were performed in the PARC K47 Corrosion Cell using both cylindrical specimens and the flat specimen holder. If one uses the Luggin Probe tip diameter which is 2.5 mm and the flat corroding sample diameter of 10.7 mm, and if the separation of Luggin probe and sample is 1 mm, one determines the cell constant, K, to be 204 cm. Given the equation

$$\Lambda = K\,\sigma \qquad (3.3)$$

where Λ is conductivity, K is the cell constant and σ is the specific conductivity, the resistance R_s is given by

$$R_s = 1/K\,\sigma \qquad (3.4)$$

Thus, if a low conductivity solution with $\Omega = 10^{-6}\Omega^{-1}\ cm^{-1}$ were put into this cell, it can be shown that R_s = 4900Ω. Accordingly, discrete values of resistance between 9.97Ω and 9990Ω were chosen to represent an appropriate range of R_s in the simulated electrodes.

A listing of the component values of the various simulated electrodes is given in Table 3-1. As can be seen from the table, the simulated electrodes fall into two groups, one in which R_p is 996Ω and R_s varies between 9.97 and 9999Ω, and the second in which R_p is 49.7Ω and R_s varies as above. The simulated electrode experiments were first run by applying the BLWN voltage signal directly across the three-component network and measuring the current response as the voltage drops across R_s rather than through the potentiostat. See Figure 3.6. The reason for this procedure was to demonstrate the best capability of the signal analyzer without the potentially complicating influences of the potentiostat. Experiments with a single simulated electrode comparing operation of the AC system with and without the potentiostat and optical isolators and as a function of signal amplitude are described in Appendix C. These experiments demonstrate that the signal-to-noise ratio is adversely affected by the presence of the potentiostat but that the character of the data plots is not changed.

Typical data illustrating all five graphical formats are shown in Figures 3.7 - 3.9 for simulated electrode "G". Theoretical curves, as depicted by the solid lines, are determined by using the calibrated component values for simulated electrode G to solve Equations (2.13) and (2.14) for the real and imaginary components of impedance. In each of the five plots of Figures 3.7 - 3.9, one observes deviation of the experimental data points from the theoretical curve at low values of frequency. Data scatter at low frequencies is attributed to 1/f error as discussed earlier. However, one also observes an apparent systematic deviation from the predicted behavior at low frequencies.

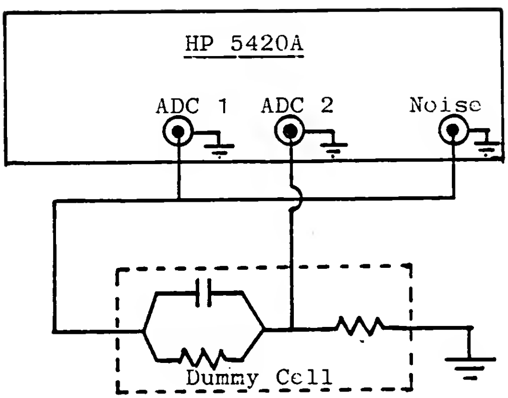
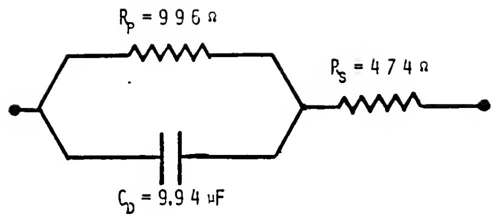


Figure 3.6 Schematic illustration of three-element network connections to the AC-system.

Table 3-1 Component Values of Simulated Electrodes

Identifier	R_s (Ω)	R_p (Ω)	C_D (μF)	R_s/R_p
D	9.97	996	9.94	10.0×10^{-3}
E	49.7	996	9.94	49.9×10^{-3}
F	100.8	996	9.94	101×10^{-3}
G	474.0	996	9.94	476×10^{-3}
H	989.0	996	9.94	992×10^{-3}
I	4520.0	996	9.94	4.54
J	9990.0	996	9.94	10.0
N	9.97	49.7	9.94	201×10^{-3}
O	49.6	49.7	9.94	998×10^{-3}
P	100.8	49.7	9.94	2.03
Q	474.0	49.7	9.94	9.54
R	991.0	49.7	9.94	19.3
S	4520.0	49.7	9.94	90.9
T	9990.0	49.7	9.94	201

(A)



COMPLEX PLANE ANALYSIS

(B)

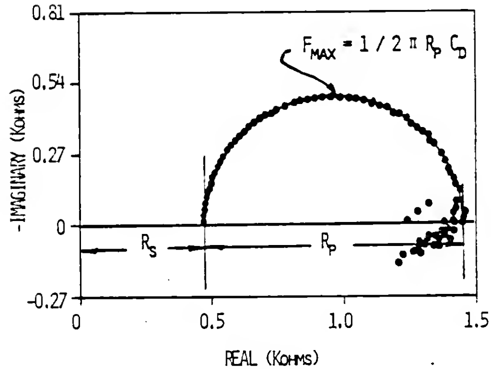


Figure 3.7 (A) Simulated electrochemical interface consisting of discrete electronic components with the values shown. (B) Complex plane plot of the impedance of the network shown in (A). Points represent values at discrete frequencies as determined by the FFT; solid line depicts analytically predicted relationship.

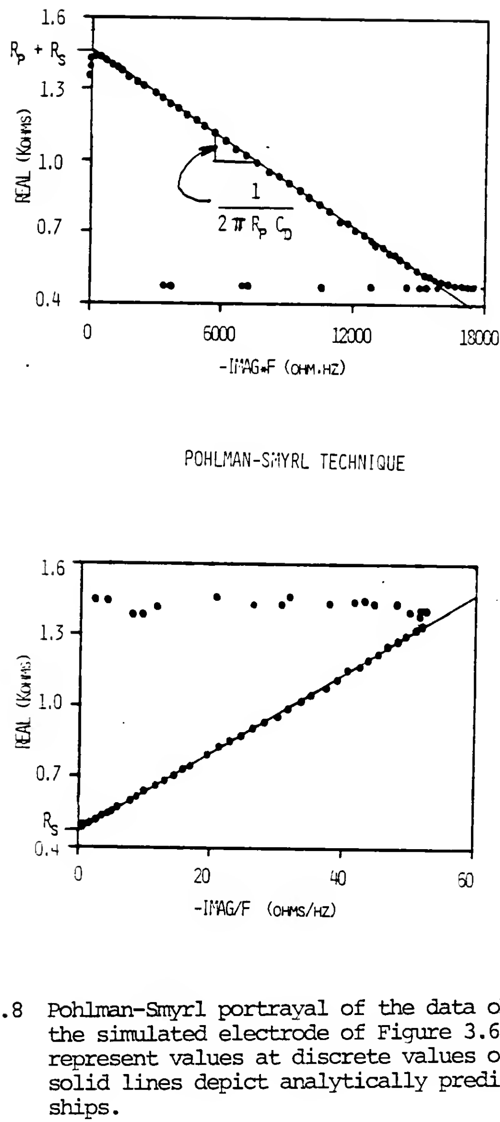
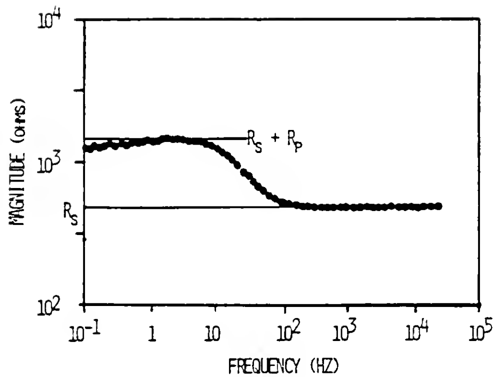


Figure 3.8 Pohlman-Smyrl portrayal of the data obtained from the simulated electrode of Figure 3.6 (A). Points represent values at discrete values of frequency; solid lines depict analytically predicted relationships.



BODE ANALYSIS

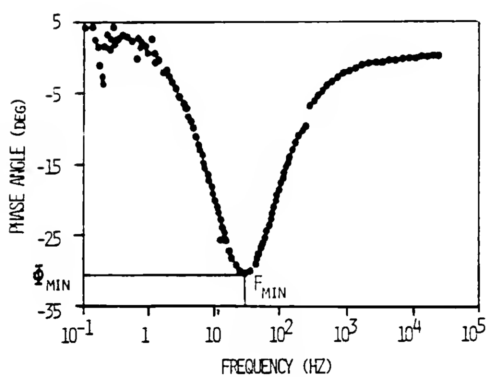


Figure 3.9 Bode portrayal of impedance data for the network of Figure 2.6 (A).

In the complex plane plot, this deviation is manifested with experimentally determined points having lower absolute values of real and imaginary components than theoretically predicted. Even more striking is the fact that the experimentally-determined impedance takes on positive imaginary values at low frequencies thus exhibiting the characteristics of a network containing inductance, which this circuit does not contain.

If we now consider the complex plane plots of other dummy cells (Figures 3.10 and 3.11), we can observe the behavior of this apparent systematic error as the values of the network resistances are changed. The series of plots in Figure 3.10 exhibits the effect of increasing the series resistance, R_s, from 9.97Ω to 9990Ω while R_p and C_D are held at 996Ω and 9.94 μF, respectively. In the series of Figure 3.11, R_s is again varied from 9.97Ω to 9990Ω while C_D is maintained at 9.94 μF and R_p is held at 49.7Ω. Since the diameter of the semi-circle is determined only by R_p, it remains constant throughout the series. The frequency at the apex of the semicircle is shown by Equation (2.17) to be a function only of R_p and C_D and therefore also remains constant throughout the series. The only supposed effect of changing R_s is to shift the constant diameter semicircle along the real axis. In these series, this is accomplished by selecting the appropriate range of the abscissa without changing the scale.

To explain the behavior of the complex plane plot series in Figure 3.10, one must be aware that C_D acts as a short circuit bypassing R_p at high frequencies and as an open circuit at low frequencies. At high

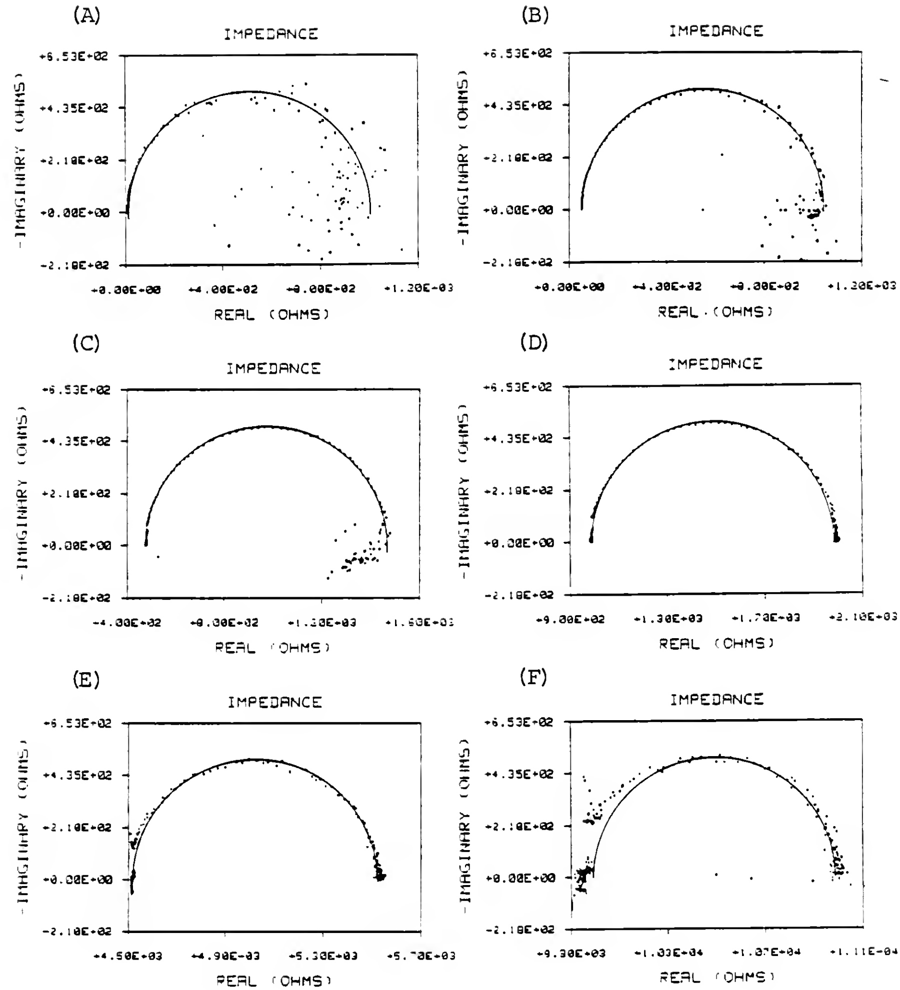


Figure 3.10 Complex plane plots of simulated electrode impedance, $R_p = 996\Omega$
(A) Dummy Cell D (B) Dummy Cell E (C) Dummy Cell G
(D) Dummy Cell H (E) Dummy Cell I (F) Dummy Cell J

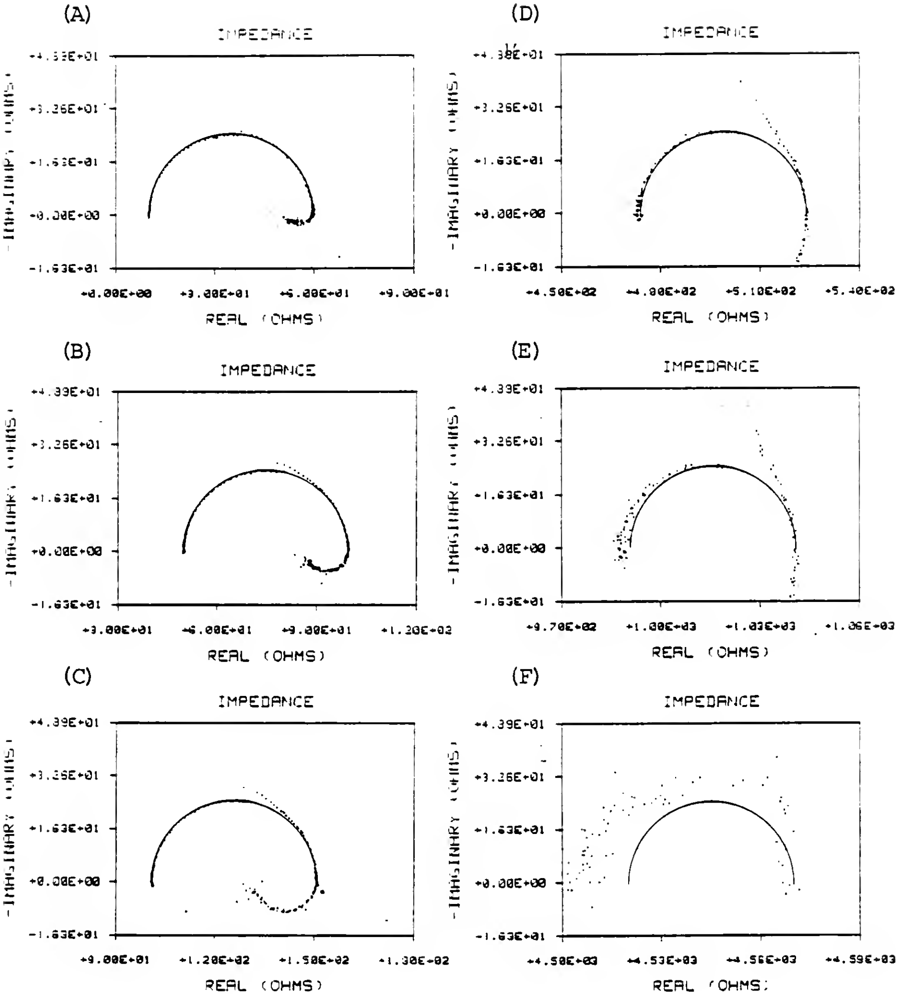


Figure 3.11 Complex plane plots of simulated electrode impedance, $R_p = 9.97\Omega$
(A) Dummy Cell N (B) Dummy Cell O (C) Dummy Cell P
(D) Dummy Cell Q (E) Dummy Cell R (F) Dummy Cell S

frequencies, therefore, the only impedance in the circuit is due to R_s. Since the rms voltage drop across the circuit is set at a level which minimizes the amplitude quantization error, the quantization error in the current measurement is also minimized. The high frequency data are therefore least susceptible to data scatter due to this error source. At low frequencies, the voltage drop across the network is divided between R_s and R_p. When R_s is small with respect to R_p, the voltage drop across R_s (used to measure current flow through the network) is small and subject to amplitude quantization error. This explains why the data scatter increases as a function of frequency when R_s/R_p is small.

The apparent systematic error present at low frequencies in Figure 3.7 may, in fact, be a manifestation of the low frequency quantization error described in the previous paragraph. When viewed as part of the plot series of Figure 3.10, it can be seen that this error disappears when R_s/R_p becomes greater than 1. There is definitely a systematic error at low frequencies in the plot series of Figure 3.11, however. The apparent inductive character of the network as evidenced by the positive-going loop is also apparently affected by the ratio of R_s/R_p. In the absence of a more plausible explanation for its occurrence, the loop is presently considered an anomalous characteristic of the 49.7Ω resistor. However, its presence indicates that there is evidence, even in dummy cells, for the so-called "mysterious inductive loops" mentioned by Mansfeld which occur in many electrochemical interfaces.

The magnitude of R_s has another effect on the results. As shown in Figure 3.6, the voltage drop across R_s is used to determine the

current flow through the network which is used in turn to determine the transfer function. The magnitude of the transfer function determined by the analyzer is thus smaller than the impedance by a factor of the resistance, R_s. Since the transfer function has to be scaled by the values of R_s to determine the impedance, any scatter in the transfer function data is also scaled by the same factor. This amplification of scatter as a function of scaling factor (magnitude of R_s) is evident in both plot series.

Another striking feature of both Figures 3.10 and 3.11 is the discontinuity in the semicircular character present in the data at frequencies immediately below 256 Hz. This gap is attributed to instrumental artifact due to the characteristic of a low-pass filter. As described in Appendix A, low pass filters are used to condition the analog signal to prevent "aliasing" with the sampling signal. In the HP5420A, either one of two filters is used depending on the selected bandwidth of analysis. The filter handling the lower frequency analysis ranges is employed when the range of analysis goes to 256 Hz or below. Further credibility is given to this argument when one observes that the semicircular character is restored at frequencies well below 256 Hz. Rather than being a fault of the individual instrument, this behavior was also observed in independent measurements made on dummy cells at Dow Chemical Company (55).

Figures 3.8 and 3.9 are portrayals of the data of Figure 3.7 in the Pohlman-Smyrl and Bode formats, respectively. In the Pohlman-Smyrl plots, one observes scatter of both the high and low frequency data in the real versus imaginary frequency portrayal. (Note horizontal

and vertical scatter of data points in Figure 3.8A.) This phenomenon is due to the fact that the imaginary component becomes zero at both high and low frequencies. Since the data are quantized, zero is never actually reached, and amplification of these near-zero values by the frequency leads to this scatter. This argument is substantiated by Figure 3.8 in which one observes no scatter in the high frequency data (in which quantization error has been attenuated by dividing by frequency); however, the quantization error of the low frequency data has been amplified by dividing by frequencies less than 1.

The Bode analysis of Figure 3.9 shows conclusively that the low frequency data suffer from a systematic rather than a random error. The cause of this deviation of the experimental data from theoretical predictions is assumed to be due to the actual low frequency behavior of the individual components.

Summary of System Development

This chapter has dealt with the creation of an AC impedance measurement system based on digital signal analysis. Chief developmental problems involved equipment interfacing and coping with two types of error associated with digital signal processing, namely 1/f error and amplitude quantization error. Tests of the digital signal analyzer and associated equipment on three component simulated electrode networks as described her and in Appendix C verify that the signal analysis approach combined with appropriate analysis techniques can return the values of the network components with acceptable levels of accuracy. These tests also reveal various ways 1/f and amplitude quantization errors can manifest themselves.

CHAPTER IV
APPLICATION OF DIGITAL SIGNAL ANALYSIS TO A CORRODING ELECTRODE

As a demonstration that the electrode impedances, as determined by digital signal analysis techniques, can be used to characterize corroding interfaces, a series of experiments was conducted on 430 stainless steel in 1N sulfuric acid. This alloy/environment combination was chosen because it is familiar to corrosion researchers, has been thoroughly characterized with DC polarization methods and exhibits active, passive and transpassive behavior depending on the polarization potential.

The system of 430 stainless steel in 1N sulfuric acid has been selected by the ASTM to be used in Standard Recommended Practice (SRP) G 5-72 (56) as a means of checking technique and instrumentation for potentiostatic and potentiodynamic anodic polarization measurements. When subjected to either potentiostatic or potentiodynamic anodic polarization, the resultant potential versus log current plots exhibit the behavior shown in Figure 4.1. Being so well characterized, this system offers the opportunity to illustrate the sensitivity of the AC technique in distinguishing among the various interface conditions.

The experimental apparatus outlined in SRP G 5-72 for performing potentiostatic or potentiodynamic scans was used with a few exceptions. A Princeton Applied Research (PAR) K47 Corrosion Cell System was used for both the potentiodynamic scan and for the AC impedance measurements.

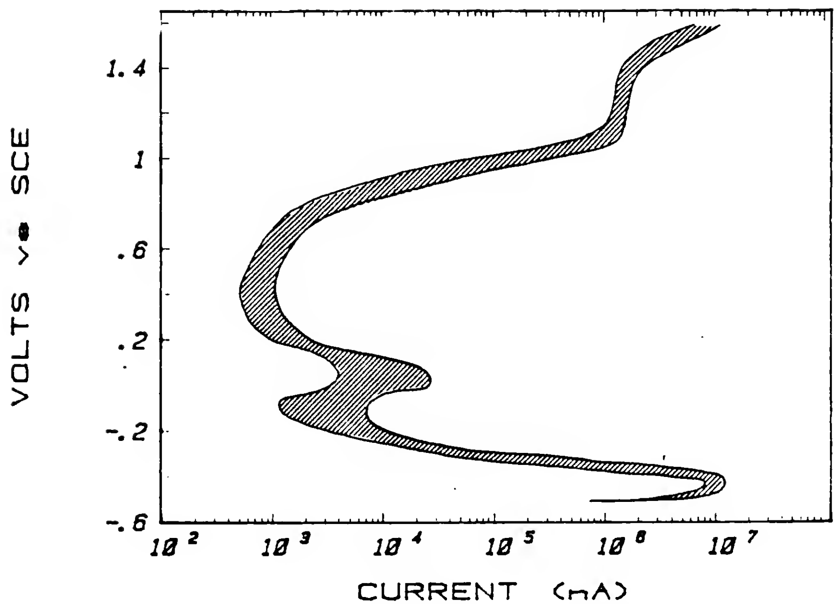


Figure 4.1 Standard potentiodynamic anodic polarization plot for type 430 stainless steel in N H_2SO_4 at 30°C with a potentiodynamic scan rate of 0.6 volts/hour. After ASTM SRP G5-72

The PAR cell system consists of a 1 liter flask equipped with ground glass ports for the test specimen, purge gas vent and entry, salt brindge/reference electrode and two high density graphite rods for counter electrodes.

The test specimen was a cylinder with a length of 0.5 in. and a diameter of 0.30 in. resulting in a total exposed surface area of 5.17 cm^2. Electrical contact to the specimen is made through a threaded steel rod within a glass tube which is compression sealed against the upper surface of the cylindrical sample. A PAR Model K77 Saturated Calomel Electrode was inserted into the salt bridge tube. Both the reference electrode and tube are terminated with a Vycor R frit, permitting ionic continuity while minimizing ionic exchange. During this operation, the bridge tube was filled with saturated KCl solution. The reference electrode was connected to the high impedance electrometer of the potentiostat.

The electrolyte was purged with nitrogen gas for at least 15 minutes prior to the insertion of the sample and purging was continued throughout the potentiodynamic run and the conduction of AC tests. The nitrogen gas was the exhaust from a 4000 SCF liquid nitrogen vessel with a purity specified at 99.99%. Since the major impurity in the liquid nitrogen is H_2O, there is no significant contribution to contamination. Because of the low heat capacity of nitrogen, there was no detectable change in solution temperature as a result of contact with the cold nitrogen gas. The cell solution was stirred constantly using a magnetic stirrer.

A potentiodynamic scan made a 1 mv/s with a PAR 350 Corrosion Measurement System is shown in Figure 4.2. The corrosion potential was

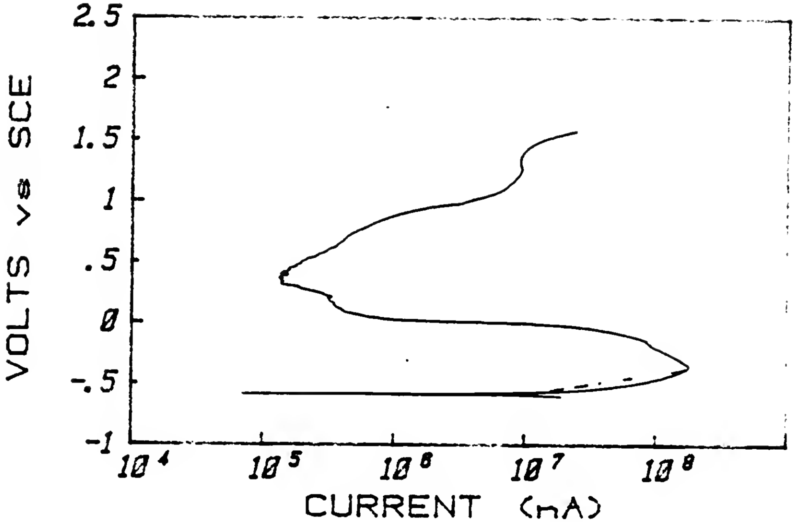


Figure 4.2 Experimentally determined potentiodynamic anodic polarization plot for type 430 stainless steel in N H_2SO_4 at 22°C with N_2 purge gas and a scan rate of 1 mV/s.

determined to be -0.571 V vs SCE with the scan being made from -0.621 V to +1.600 V. Solution resistance was determined to be 0.29Ω and polarization resistance of 1.1Ω. The potentiodynamic behavior of Figure 4.2 exhibits some differences from the reference plot of Figure 4.1. These are attributed to the differences in scan rate and the fact that nitrogen rather than hydrogen purge gas was used. Since the purpose of the AC experiments is to illustrate impedance characterization of the metal electrode under conditions of active, passive and transpassive behavior, the minor differences between Figures 4.1 and 4.2 are of no consequence.

Upon completion of the potentiodynamic scan, the specimen was repolished and placed into fresh solution. A PAR 173 potentiostat was used to set DC potentials in increments of 200 mV anodic to the corrosion potential. A BLWN signal of approximately 40 mv p-p was fed to the summing junction while the respective DC potential was maintained by the potentiostat. See the wiring schematic of Figure 4.3. The figures which follow are the complex plane plots of the impedance behavior at the respective values of anodic potential.

At the corrosion potential (Figure 4.4), the complex plane plot resembles the semicircular form of the three-element network model at intermediate frequencies but exhibits a second semicircular lobe at lower frequencies. One notes that the real axis intercept occurs at about 0.3 ohms while the uncompensated resistance measurement in the DC polarization run (Figure 4.2) listed as 0.29 ohms. Since the Luggin probe was moved between the potentiodynamic scan and the AC run, this should be regarded as good agreement for solution resistance.

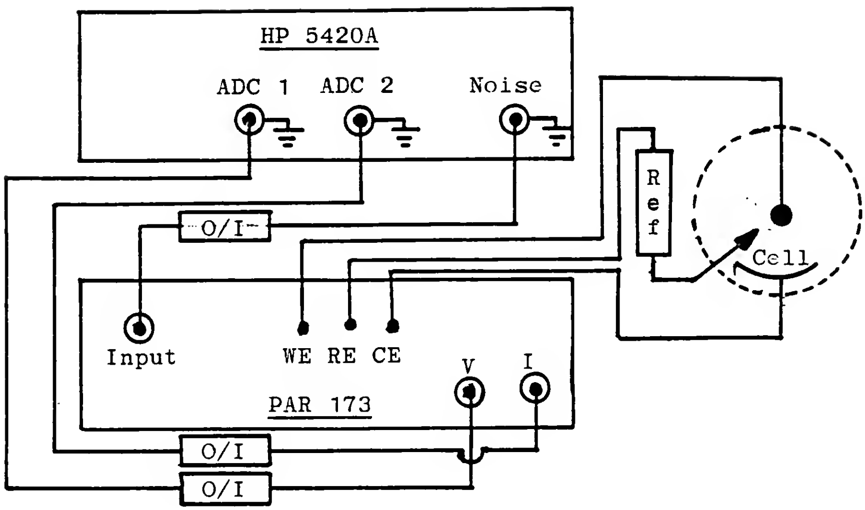


Figure 4.3 Schematic illustration of wiring for AC impedance experiments on 430 stainless steel.

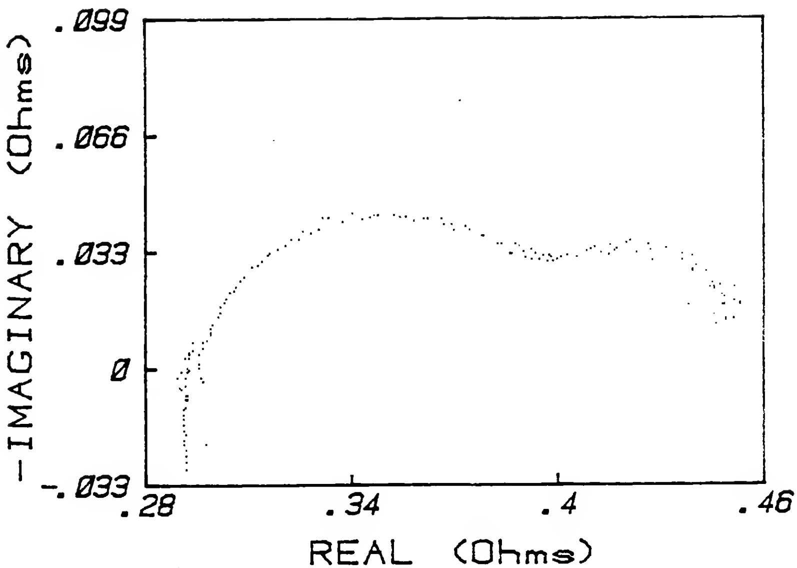


Figure 4.4 Complex plane plot of electrode impedance of 430 stainless steel in N H_2SO_4 at the corrosion potential.

If one fits a semicircle to the first lobe of Figure 4.4, the value for R_p in a three-element network model is found to be about .09 Ω, more than an order of magnitude below the polarization resistance determined by the DC polarization method.

Although the complex plane method cannot determine the value of C_D exactly unless the frequency at the apex of the semicircle is known, its value can be bracketed by viewing the range of frequency over which data were gathered. Using this method it can be shown that the value of C_D lies between 10^4 and 10^5 μF. Fittings this range of C_D values to the plot of phase angle vs frequency (see Figure 4.5) gives a good match when C_D is 10^5 μF.

To evaluate two-lobe behavior a more complicated model than the three-element network is necessary. Since a passive film is expected on this material at higher polarization potentials, it was reasoned that an extremely thin film might also exist at the corrosion potential. A network model similar to that proposed by Richardson, Wood and Breen (37) was therefore considered. See Figure 4.6. Although more complicated than the three-element network, the impedance of this model can also be evaluated analytically as a function of frequency and element values. A BASIC computer program, GRAFIT, was written to accept any value of R_p, R_s, R_F, C_D, C_F and to plot the resultant impedance over a range of frequency from 0.25 to 25000 Hz. See Appendix D. By using the values for R_s, C_D, and R_p already determined, and experimenting with various values of C_F and R_F, the fit of Figures 4.7 was achieved. Although not a perfect fit over all ranges of frequency, one may be confident that all element values are within the right order of magnitude.

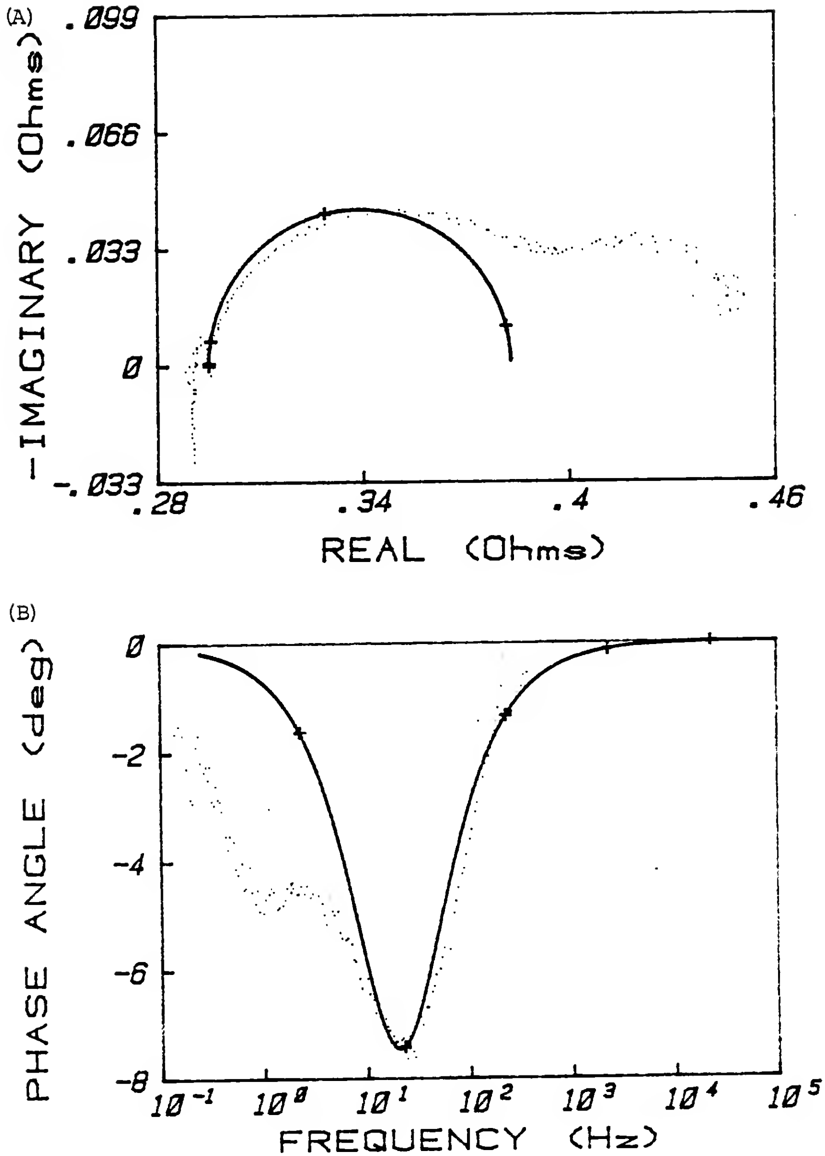


Figure 4.5 Complex plane plot, (A) and phase angle vs. frequency plot, (B) of electrode impedance of 430 stainless steel maintained at the corrosion potential. Solid lines depict fit of an assumed three-element network where $R_s = .295\Omega$, $R_p = .088\Omega$, and $C_D = 0.1$ F.

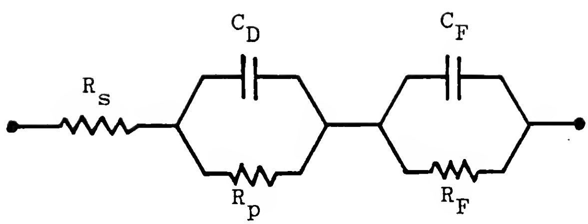


Figure 4.6 Five element network model representing the corrosion of 430 stainless steel in N H_2SO_4.

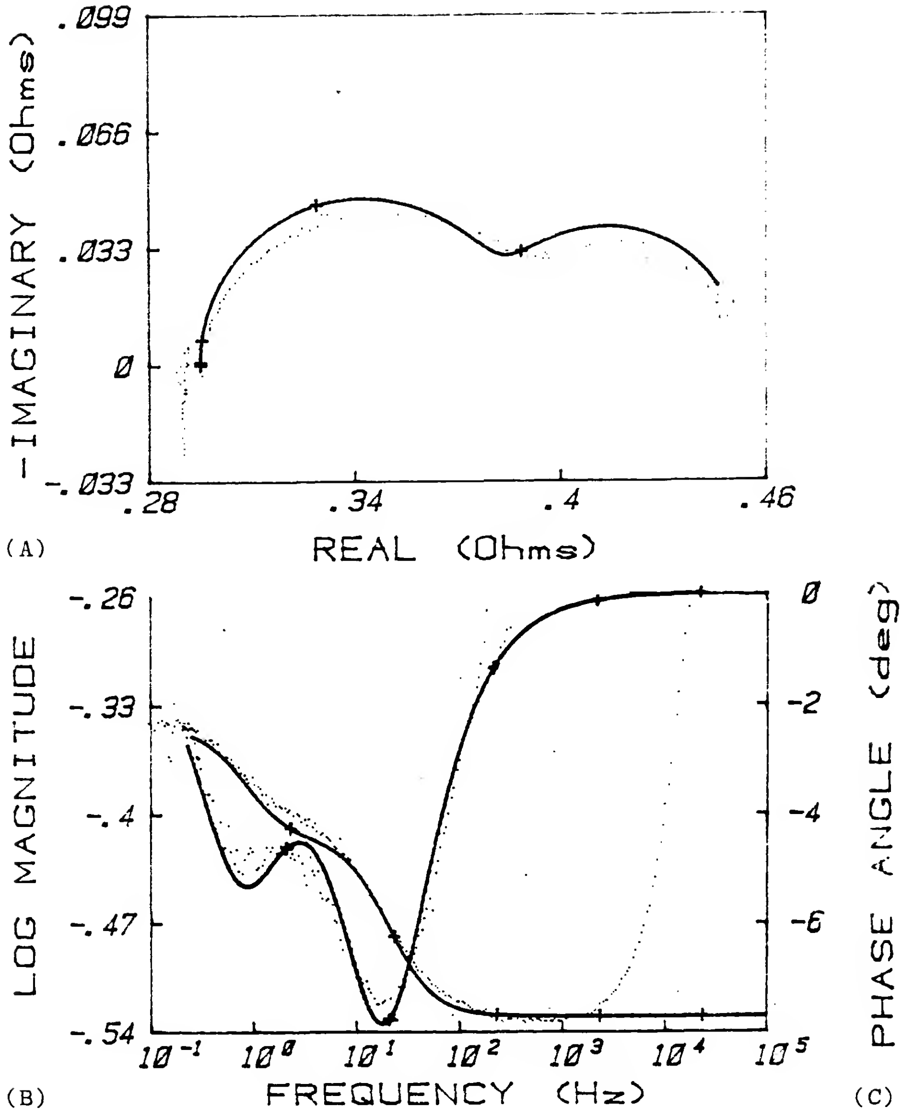


Figure 4.7 Complex plane plot, (A) and Bode plots (B) and (C) of electrode impedance of 430 stainless steel in N H_2SO_4 at the corrosion potential. Solid lines depict fit of five-element network model of Figure 4.6 where R_S = .295Ω, R_p = 0.088Ω, R_F = 0.07Ω, C_D = 0.1 F and C_F = 3 F.

The magnitudes of the element values are startling in view of both the findings of the DC polarization experiment and of accepted values of double-layer capacitance. Rationalization of these findings is deferred until after the consideration of electrode impedance at other values of DC polarization potential. Another puzzling aspect of the complex plan plot is the appearance of a positive imaginary at high frequencies. This feature of the experimental data is attributed to artifact from the optical coupling circuits as discussed in the previous chapter. Consideration of the high frequency tail in Figure 4.7 (C) also lends credibility to this explanation. The apparent data mismatch at the high frequency real axis intercept is believed to be caused by the low pass filter in the vicinity of 256 Hz, an artefact also discussed previously.

The impedance of the stainless steel in the active region is considered next. The complex plane impedance plot shown in Figure 4.8 was made at a polarization potential of about -300mv vs SCE. It can be seen that this plot is similar to the one made at the corrosion potential except that there is much more scatter in the low frequency data which defines the second semicircular lobe. The scatter in this case is attributed to the instability of what has been modeled as a thin passive film. A reasonably good match of experiment and model is obtained with element values shown in Figure 4.9. The decrease in R_p is to be expected in the active corrosion region.

In the passive region, the behavior changes drastically as illustrated by Figure 4.10. However, the behavior can still be predicted with the same model. As shown in Figure 4.11, a reasonably

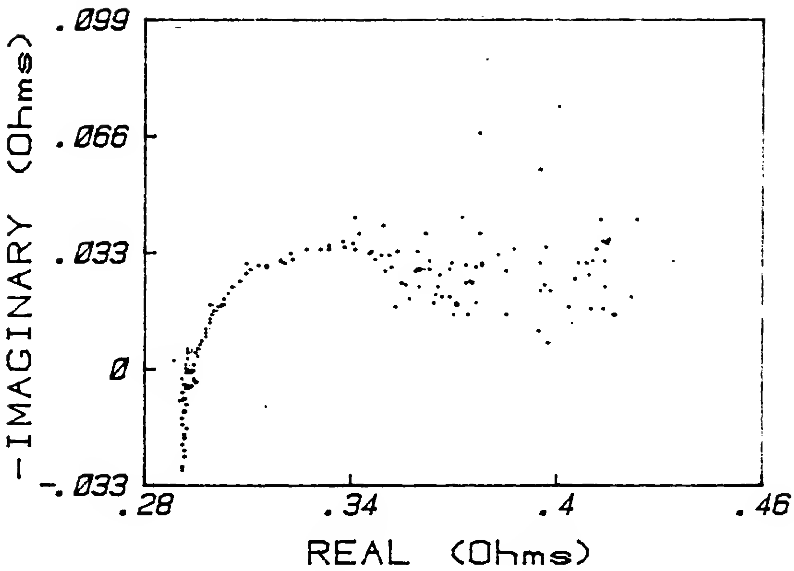
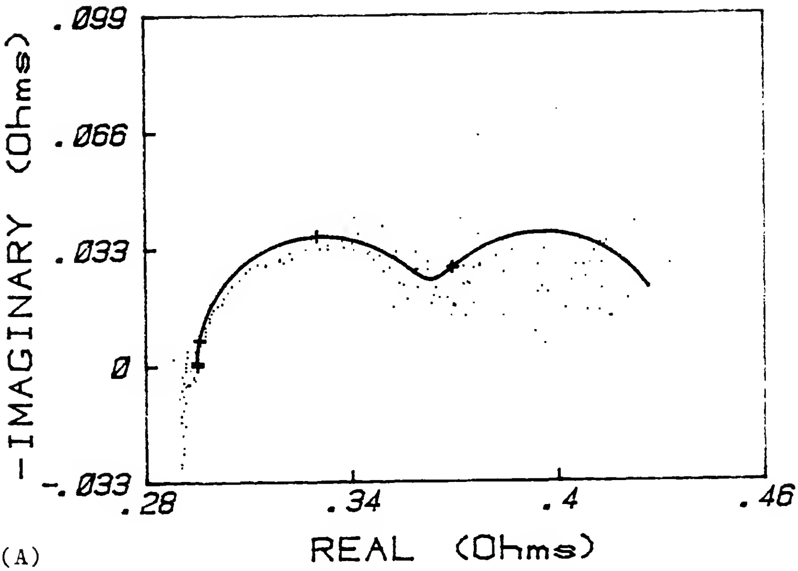
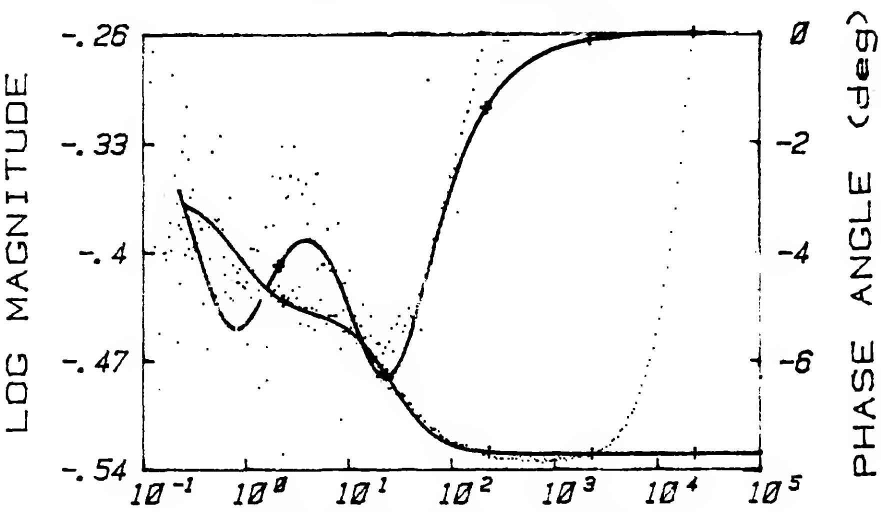


Figure 4.8 Complex plane plot of electrode impedance of 430 stainless steel in N H_2SO_4 polarized into the active corrosion region at a polarization potential of -300 mV vs. SCE.



(A)



(B) (C)

Figure 4.9 Complex plane plot (A) and Bode plots (B) and (C) of electrode impedance of 430 stainless steel in N H_2SO_4 at -300 mV vs. SCE. Solid lines depict fit of five-element network model of Figure 4.6 where R_S = 0.295Ω, R_P = 0.1Ω, R_F = 0.07Ω, C_D = 0.1 F, and C_F = 3 F.

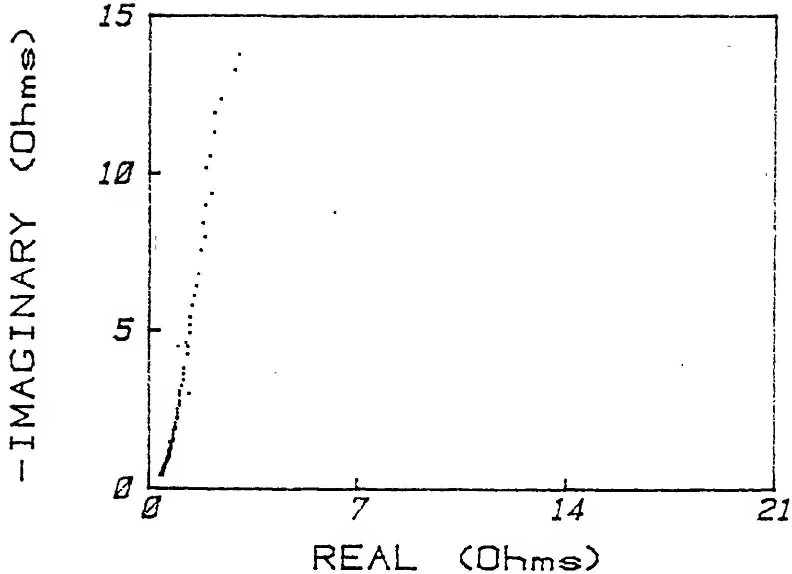


Figure 4.10 Complex plane plot of electrode impedance of 430 stainless steel in N H_2SO_4 polarized into the passive region at a polarization potential of +450 mV vs. SCE.

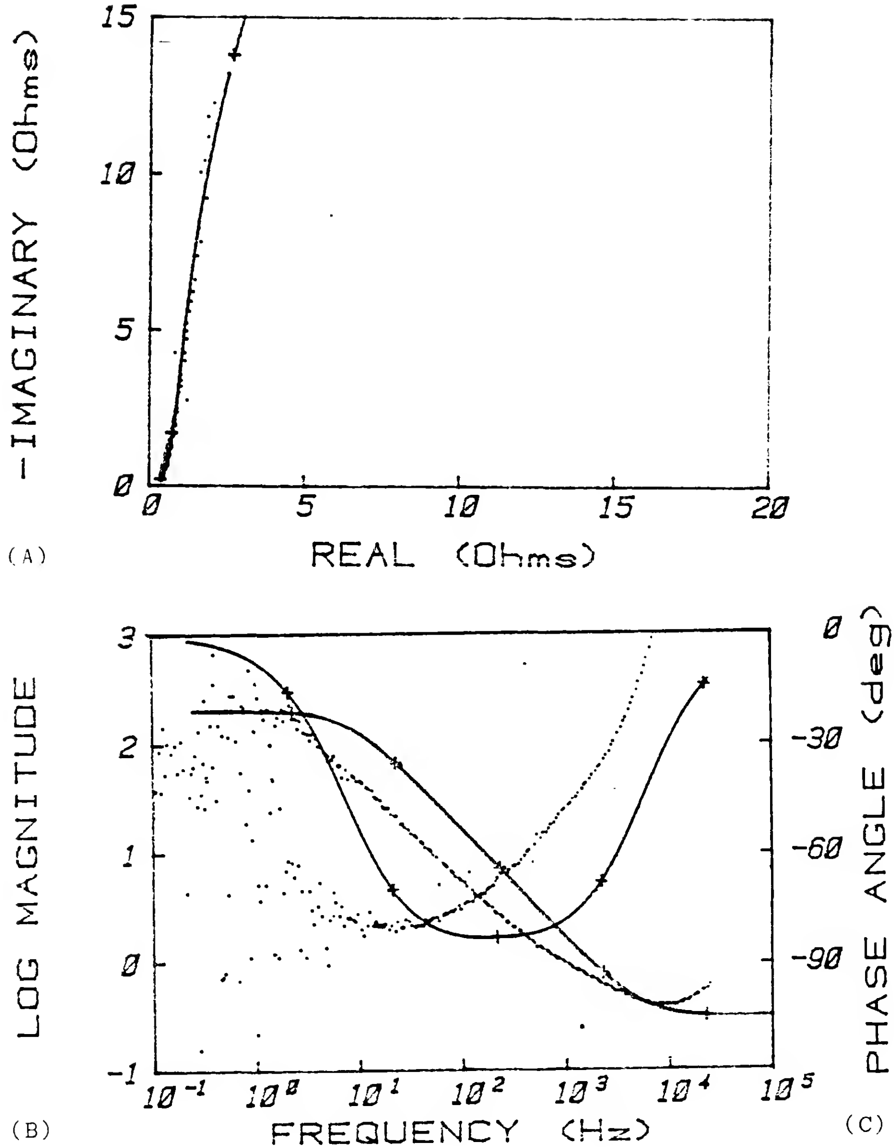


Figure 4.11 Complex plane plot (A) and Bode plots (B) and (C) of electrode impedance of 430 stainless steel in N H_2SO_4 at +450 mV vs. SCE. Solid lines depict fit of five-element network where C_D = 1000μF, C_F = 100μF, R_p = 0.6Ω, R_f = 200Ω, and R_s = 0.3Ω.

good match is obtained by increasing R_p slightly, increasing R_F dramatically and decreasing both C_D and C_F.

Figures 4.12 and 4.13 illustrate how the model can also be used to describe transpassive behavior. Here one notes a dramatic decrease in the values of R_f indicating film breakdown.

The difference in the impedance descriptions of a corroding metal electrode at the corrosion potential (as contrasted with the active, passive and transpassive potentials) illustrates the sensitivity of the AC impedance method to changes in the surface conditions of the interface and especially to changes in the protective character of a passive film. While the changes in the resistance of a film are intuitive and may be surmised from DC polarization data, the capacitive character of the interface cannot be determined by direct current methods.

The data on capacitance provide additional descriptive information about the nature of the electrochemical interface at a particular instant in time. The fact that capacitance values of a corroding electrode exhibit drastic variations depending on polarization potential and the fact that they differ from accepted values of double-layer capacitance of well-behaved systems on dropping mercury electrodes should be considered as further evidence of the sensitivity of this method to the condition of surface.

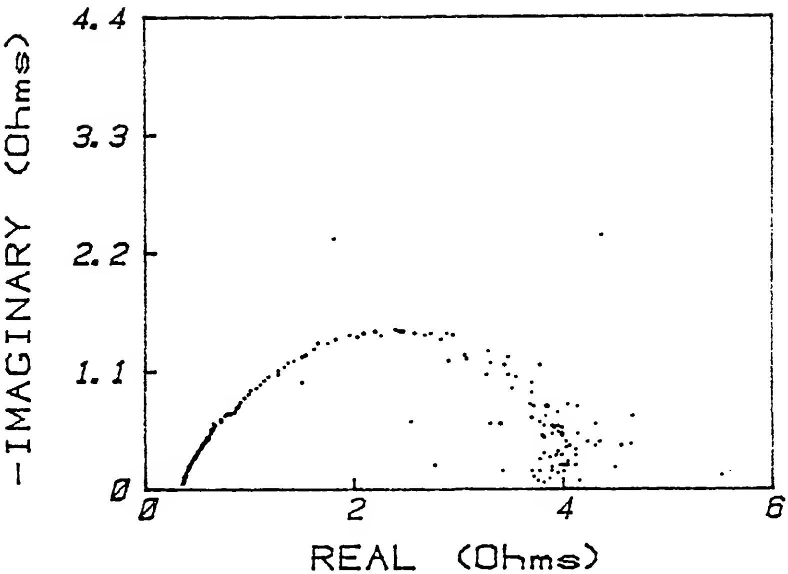


Figure 4.12 Complex plane plot of electrode impedance of 430 stainless steel in N H_2SO_4 polarized into the transpassive region at a polarization potential of 1.300 V vs. SCE.

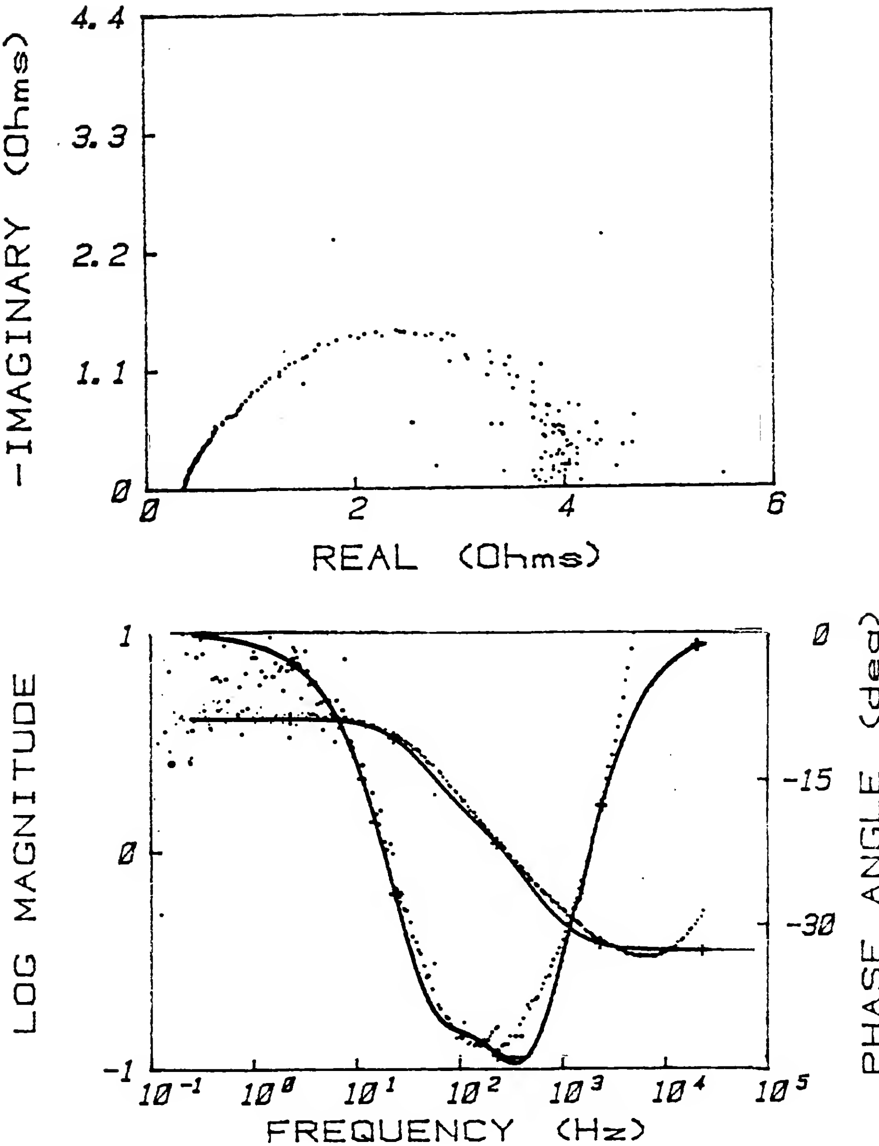


Figure 4.13 Complex plane plot (A) and Bode plots (B) and (C) of electrode impedance of 430 stainless steel in N H_2SO_4 at a polarization potential of 1.300 v vs. SCE.

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE RESEARCH

Recapitulation: Objectives and Premises

The specific objectives of this work as stated in Chapter I were (1) to assemble an in situ corrosion monitoring system based on digital signal analysis using off-the-shelf commercially available electronic equipment; and (2) to demonstrate the capabilities and limitations of such a system in characterizing the corrosion of 430 stainless steel in 1N sulfuric acid. This attempt is based on the premise that a corroding interface, whether active, passive or transpassive, behaves as a relaxed, linear time-invariant system with respect to dynamic response to voltage perturbations.

The extent to which these assumptions are valid was considered in some detail. Relaxedness is probably a reasonable assumption if the perturbation signal amplitude is kept above the level of random system noise. Although no electrochemical system is truly linear over an extremely broad range of voltage, behavior over small enough voltage excursions can be considered linear. Time invariance is also not rigorously valid for a corroding interface since passive films may grow slowly or change their character over time. However, if the period of measurement is short, on the order of minutes, the assumption of time invariance is also reasonable.

In comparing the proposed corrosion monitoring system based on digital signal analysis with the more familiar DC polarization

techniques, the following facts and observations are apropos. DC methods yield a corrosion current which is related to a corrosion rate. There is good correlation between this electrochemically-determined rate and weight-loss data for systems corroding actively and uniformly over the entire surface. However, the corrosion rate determined by DC techniques is not a reliable indicator of corrosion inhibitors. In such systems, which represent the majority of materials which exhibit some form of inherent corrosion resistance, one seeks a parameter or combination of parameters which characterizes the state of the corroding electrode in its environment. While polarization resistance or corrosion current is such a parameter, it does not by itself give an accurate picture of the state of the surface. Furthermore, the imposition of a DC voltage signal on the interface unquestionably alters the state of the corroding electrode in some way, if only by changing the concentration of adsorbed ions.

Alternating current techniques offer the capability of being able to determine capacitive as well as resistive properties of the interface. While there is no apriori guarantee that the capacitance values will be enough to completely characterize the interface, they are, nevertheless, additional information about the corroding electrode. In addition, alternating current voltage signals of appropriate amplitude and of high enough frequency will have little or no permanent impact on the state of the interface.

Motivated by this logic, one seeks a method of determining the frequency response of a corroding electrode to an alternating current perturbation. One frequently-used technique is to perturb the

interface with a single frequency at a time, using a lock-in amplifier to determine the response signal at that same frequency. This approach provides a high signal-to-noise ratio, but has the disadvantage of being time-consuming, a factor which may have impact on the assumption of time invariance mentioned previously.

In principle, the digital signal analysis approach overcomes all of these difficulties. A time-dependent signal is digitized into a N-sample ensemble whick is converted into an N-channel frequency spectrum via the FFT algorithm. In the frequency domain, the perturbation and response spectra may be manipulated algebraically to yield the transfer function, or in this case, the impedance of the electrochemical interface. Analysis of the impedance may then yield values of equivalent circuit elements.

None of the aforementioned concepts are new or unique to the fields of electrochemistry or corrosion. Epelboin and coworkers have been prolific in reporting experiements on the impedance of corroding electrodes, primarily with sequential measurements of incrementally-varied frequencies. Smith and coworkers have used FFT faradaic admittance measurements to study the kinetics of electrochemical reactions of the dropping mercury electrode. Pryor and coworkers have examined anodic films at several different frequencies. The work described here represents the first known attempt to use off-the-shelf commercially available digital signal analysis equipment to measure, in situ, the impedance of and to characterize the state of the corroding interface of 430 stainless stell in 1N H_2SO_4 in the active, passive and transpassive ranges of polarization potential.

Capabilities and Limitations of Digital Signal Analysis

The equipment assembled to evaluate the impedance of a corroding electrode was first tested on a three-element dummy circuit which represented the simplest model of an electrochemical interface. By varying the element values of the equivalent circuit one quickly became acquainted with some of the difficulties inherent in this method. Despite the use of 12-bit A/D converters, digitization error became a problem if the measured voltage became a small percentage of the full scale value. Since the instrument was not equipped with an autoranging capability it was recognized that the perturbation and response voltage signals would have to be monitored and appropriate ranges selected manually during the course of an experiment in order to assure acceptable S/N.

Another concern was 1/f error. Since the N frequencies produced by the FFT are equally spaced, the percent uncertainty at a particular frequency is inversely proportional to that frequency. Since this introduced unacceptable error beyond one decade of analysis, it was decided to tie consecutively-gathered spectra together, discarding low resolution overlapping data. The resultant five-decade frequency spectrum was thus assured of a minimal 1/f errof. However, the required five sequential measurements compromise one of the original purposes for favoring FFT analysis over lock-in amplifier measurements, namely real-time capability. In its present configuration, data is collected over a nine-minute period.

Another outcome of the dummy electrode studies was the discovery of an apparent oversight in the design of the HP5420 analyzer, a low-pass

filter with premature cutoff. This behavior manifested itself as a deviation from theoretical predictions or real and imaginary components of impedance at frequencies immediately below 256 Hz. From this observation of dummy electrode behavior, the author was spared trying to explain the significance of instrumental artefact in the impedance of real electrodes. The dummy electrode experiments also illustrated the high frequency behavior of the optical isolator circuits and the associated deviation from theoretical predictions. Apart from these idiosyncrasies of the AC system, the dummy electrode studies also illustrate the system is capable of experimentally determining the values of a known three-element network reliably.

The application of the AC system to the study of corrosion of 430 stainless steel in 1N H_2SO_4 revealed more capabilities and limitations. The data analysis capabilities of the system software were conceived based on the three-element network model. The two-lobe behavior in the complex plane plot at the corrosion potential revealed that a more complicated equivalent circuit model was required. The chosen model is actually a straightforward extension of the three-element network assuming the presence of a film in series with the electrochemical interface, a film possessing both resistive and capacitive character. It is interesting that this five-element network model appears to be adequate at the corrosion potential and at all intermediate potential ranges up to and including transpassive behavior. The values of capacitance at the corrosion potential are startling in magnitude, but their variation as a function of potential

is ample proof of the sensitivity of this method to the character of the corroding interface. The fact that a single equivalent circuit model can be used at all values of polarization potential and the reasonable variation of resistance values also support the capability of this technique in characterizing the corroding electrode with equivalent circuit values.

If quantification of equivalent circuit elements proves to be the most effective way of characterizing an interface, the analytical determination of the values from the impedance data must be improved upon. The present trial and error method is time-consuming and awkward and certainly lacks mathematical elegance. Other methods of characterizing protective film behavior should be sought, possibly expressing it as a mathematical combination of several network parameters.

In this set of experiments, the upper frequency instrumental artefact due to the optical isolator circuits did not appear to jeopardize the determination of any network parameters. However, it certainly diminishes the system capability below the desired five-decade frequency range. Attempts should be made to either improve the frequency response of the optical couplers or to find a simpler method of integrating equipment components.

Recommendations for Future Research

Future work in this field should be divided into three categories: system refinement, analytical methods development and corroding electrode studies.

System Refinement

Minimizing discretization error. Although experiments with dummy networks and real corroding electrodes demonstrate the system's capability to determine element values of equivalent circuits, it is still a somewhat awkward system to use. The 100mv lowest scale limitation of the A/D converters intensifies the problem of discretization error when the desirable perturbation signal amplitude of 2-5mv is used. The addition of selectable low-noise 10x or 100x amplifier stages prior to the A/D converters would be a welcome addition. Since the network impedance of an electrochemical interface can vary by orders of magnitude as a function of frequency, however, this addition would not totally solve S/N problems due to discretization error. Autoscaling might offer a solution if scale changes could be tracked and used to scale the frequency domain data.

The problem of digitizing an analog voltage signal whose amplitude may vary over several orders of magnitude is one shared by those who code and decode voice message signals in telecommunication syatems. In this case coding corresponds to an A/D step; decoding corresponds to a D/A step. Where voice signals are coded and decoded uniformly, or in even digital steps, weak signals experience much more distortion than strong signals. Bell Telephone engineers have overcome this problem with a coding-decoding process called companding (57).

A compandor is a non-uniform coder-decoder pair which performs a logarithmic compression on the analog voice signal, a uniform digitization of this distorted signal and logarithmic expansion during decoding. It is clear that such a method can provide a greatly

improved S/N at low signal smplitudes. Obviously the transformation of a distorted time domain signal results in a distorted frequency domain signal. However, since the FFT is a linear operation, the distorted frequency domain signal can be logarithmically expanded in the same manner as a time domain signal. Companding of analog inputs should be considered as a means of minimizing the effect of discretization error at low signal amplitudes.

Minimizing 1/f error. As mentioned previously, 1/f error contributed to a significant compromise in the utilization of the digital signal analysis technique. One of the touted advantages of the FFT approach is that a time domain signal can be analyzed for 256 discrete frequencies simultaneously rather than sequentially at discrete frequencies. However, due to 1/f error, the desired five-decade analysis range could be achieved with acceptable error only by performing five sequential analyses over decade-varying frequency ranges. The resultant 1150+ data points were more than adequate to define the impedance behavior over the full range but suffered from the disadvantage that the data were collected at five discrete intervals of time.

One method of minimizing the 1/f error would be to use an FFT which evaluates over a higher number of discrete intervals, say 1024 or 2048. However, a 2048-interval analyzer would still acquire less than two decades of frequency data at the same level of error while increasing the computation burden and computation time of the FFT by a factor of 64.

In reality, 250 data points would be adequate to define the impedance if they were uniformly distributed over the five decades.

i.e. if the log frequency scale were divided into 250 evenly-spaced intervals. This method was in fact used in the numerical-analytical determination of the impedance of the five-component network. (See the description of GRAFIT in Appendix D.)

How this might be implemented into FFT analysis requires thorough consideration of the method by which the FFT is determined. One seeks to distribute the frequency intervals evenly over the log frequency scale. Since the FFT is a linear operator it would appear that distribution of the time domain samples evenly over the log time scale would be the only requirement. Time precludes rigorous proof of this hypothesis but if it is correct, one need only sample the time domain data in logarithmically distributed intervals. This would require a sampling signal with the pulses appropriately spaced. Whether this or another method is used, effort should be made to perform the five decade analysis using a single 256 sample ensemble of data.

Perturbation signals. The white noise signal used to perturb the electrochemical interface was used because of its availability rather than being selected as an ideal signal source. Other investigators have used single frequency sinusoidal signals or combinations of particular frequencies or pseudo-random noise signals. One type of signal which would appear to be ideal for evaluating any relaxed, linear, time-invariant system would be the impulse function. Since the Fourier transform of the impulse function is identically unity, the measurement of the cross power spectrum of the system response should be exactly equal to the transfer function of the system.

Despite the fact that Smith's empirical comparison of the impulse function with other perturbation signals seems to indicate that impulse is a "less efficient" signal than others, there has really been no sound theoretical basis for excluding it. Further investigation should be made of the utility of various signal types as perturbation sources.

Analytical Methods

Equivalent circuit analysis. The graphical analysis of all five plot types to determine equivalent circuit values for a three-element network is straightforward and fully within the capability of the main program software. At the present time the analysis of a five-element network can be done only by trial and error. This is an obvious area of future development in the category of analytical methods. One of the disadvantages of the equivalent circuit method of analysis is that these analysis methods must be modified each time a new interface model is proposed. Further investigation should be made of alternative methods to characterize the corroding interface using dynamic response data.

Application of Cohtrol Theory to corroding interfaces. An entirely different analytical approach should also be attempted. If one considers a plot of current vs potential for a metal which exhibits a region of passivity, there is a region of constant low current in the range of polarization potentials in which the metal is protected by the passive film. In this region the electrochemical inteface appears to be behaving as a negative feedback control system, in this case as a current regulator. Increases in potential have the effect of

increasing film thickness which in turn limits current flow. This "self-healing" nature of the film in the passive region is exactly the quality of a metal surface which renders an otherwise active metal corrosion-resistant.

One of the most pertinent questions one can ask about any control system concerns its stability. Stability, in this context, means its ability to restore the set current when perturbed from equilibrium, i.e. changing the potential. In this case, changing the potential also gives the control system different characteristics, since the potential is intimately associated with the thermodynamic state of the passive film.

The problem is a complicated one; however, control theory, provides methods of determining whether a system is stable or unstable. If one considers the Bode portrayal of the so-called open-loop transfer function of the system one can determine graphically whether the control system is stable or unstable. It is logical to assume that a control system consisting of an electrochemical interface and a passive film would have its stability dependent on potential and that anodic protection could be effected by maintaining the potential at the value at which the system was most stable. Although such an approach avoids consideration of electrochemical mechanisms, it woulc be an effective way of determining how to control an anodic protection system.

Corroding Electrode Studies

Gathering more data on real corroding electrodes will do much to enhance the credibility of this technique, and to better identify

its specific strengths and weaknesses. For example, can this technique with some or all of the refinements described earlier be used to characterize the state of the corroding electrode for which the solution resistance is orders of magnitude larger than the other resistive components? Examples of this type of problem are the evaluation of the corrosion of steel reinforcing rods buried in concrete or the deterioration of metals in weakly conducting electrolytes such as alcohols.

Can the method be used to evaluate the character of passive films on aluminum or other valve metals? These films are thick and can be made thicker through anodization. Would this approach be an appropriate quality control for the anodization process? Can it be used to monitor time-dependent changes or an anodic film in a particular solution?

Can this technique be used to evaluate the corrosion protection quality of other types of coating such as paints? If the coating is porous will this technique be able to distinguish between this and a properly applied coating? What can this technique reveal about inhibitors in a coating material or in the solution?

It may also be fruitful to compare this technique one-on-one with the sequential frequency data collection techniques. Is the FFT method significantly faster than sequentially-gathered data? Does the lower S/N ratio of the lock-in amplifier approach make it more reliable than the FFT method? Can a frequency be identified below which the character of the corroding system is changed, i.e. no longer behaves with the same transfer function as at higher frequencies?

This application of digital signal analysis technology to the study of corroding electrodes illustrates that the electrochemical interface exhibits behavior analogous to that of virtually all dynamic systems when perturbed by white noise. The frequency response is a unique fingerprint of its character, a character revealed only partially and with much more effort by direct current methods. While many more investigations must be performed to address the questions posed on the previous pages and other data analysis techniques required to reveal all of the corrosion-specific information contained in the transfer function, the method offers significant promise to become the technical basis of real-time corrosion-monitoring systems. This work represents a small step toward the technical realization of automatic, rapid, accurate and inexpensive characterization of corroding interfaces.

APPENDIX A
DIGITAL SIGNAL ANALYSIS TERMINOLOGY

A.1 Introduction

The motivation for performing digital signal analysis lies in the premise that the response of a system to a time varying perturbation will reveal information about the dynamic character of the system which may not be obtainable by other means. This approach is used widely in mechanical vibration testing, acoustics and in the analysis of feedback control system stability. Sophisticated equipment has been developed to perform this frequency response analysis and is capable of a wide range of data collection functions in addition to the determination of system transfer functions. The application of these sophisticated signal analysis techniques to electroanalytical chemistry in general and to corrosion processes in particular is recent and represents a novel and potentially fruitful method of corrosion monitoring. In this section, the theoretical basis for digital signal analysis is discussed briefly followed by a description of the features of the analysis equipment used for this work (58, 59).

A.2 Time-to-Frequency Transformation

Fundamental to the execution of signal analysis is the conversion of the perturbation and system response into time domain electrical signals, i.e., voltage versus time. In mechanical vibration testing, this may be accomplished with strain gauges, force transducers, or

accelerometers; in acoustics, microphones are used. In electrochemical impedance measurements, the voltage perturbation signal is used directly and the current response is converted into a voltage signal either as the voltage drop across a resistor or through a zero resistance ammeter.

In order to relate the response of a system to the input perturbation, it is much more efficient to convert both time domain signals into the frequency domain. The rationale for performing this transformation is that frequency domain functions may be manipulated algebraically to yield frequency response information in contrast to the more complex integral/differential relationships required to analyze the time domain functions. For example, the input-output relationship of a linear system can often be described by the so-called convolution integral

$$y(t) = \int_{-\infty}^{\infty} \omega(t,\tau)x(\tau)d\tau \qquad \text{A.1}$$

where $y(t)$ is the time domain response to the input perturbation $x(t)$. The function $\omega(t,\tau)$ is called the weighting function and embodies the physical properties of the system. In the frequency domain, the input and output can be related algebraically as

$$Y(f) = H(f)\cdot X(f) \qquad \text{A.2}$$

where $Y(f)$ and $X(f)$ are the output and input, respectively, and $H(f)$ is the so-called transfer function.

The mechanism for performing this transformation from time to frequency domain is found in the single-sided Fourier transform integral,

$$F(\omega) = \int_{0}^{\infty} f(t)e^{-j\,2\pi\omega t}dt \qquad \text{A.3}$$

The form of Equation A.3 suggests that it can be evaluated by

converting the exponential $e^{-j\omega t}$ to the trigonometric form cos (ωt) - j sin (ωt), and this is the basis for the evaluation method used in computer-based analyzers. When one attempts to evaluate $F(\omega)$ analytically, however, the explicit correspondence between the time domain function and frequency domain function defined by Equation A.3 results in an expression for $F(\omega)$ which is the sum of an infinite series of sine and cosine terms. One therefore seeks a numerical solution to the evaluation of Equation A.3.

Fortunately, the discrete Fourier transform (DFT) algorithm offers a very good approximation of the Fourier integral and can be computed in a more reasonable period of time:

$$F(k\beta) = \sum_{n=0}^{n=N-1} f(n\tau)\, e^{-j\frac{2\pi(n)k}{N}} \qquad \text{A.4}$$

One implements the DFT by breaking up the time domain function $f(n\tau)$ into N samples τ seconds apart in a total data block of $N\tau$ seconds. Then one restricts the frequency domain function to a finite frequency range of interest and divides it into K discrete equally spaced elements β Hz apart. Equation A.4 yields a frequency spectrum $F(k\beta)$ composed of K discrete values from the sampled signal $f(n\tau)$. (See Figure A.1) In the case of the HP5420 signal analyzer used for this work, K = N = 256 for both the input and output signals. Equation A.4 indicates that for each of the 256 values of n, the value of $e^{-j\frac{2\pi nk}{N}}$ must be computed and multiplied by $F(n\tau)$ and that this must be repeated for each of the 256 values of k. This represents a calculation burden of over 65,000 steps. Of course, this burden is multiplied by two in order to get the same information for both input and output, resulting in an unacceptably

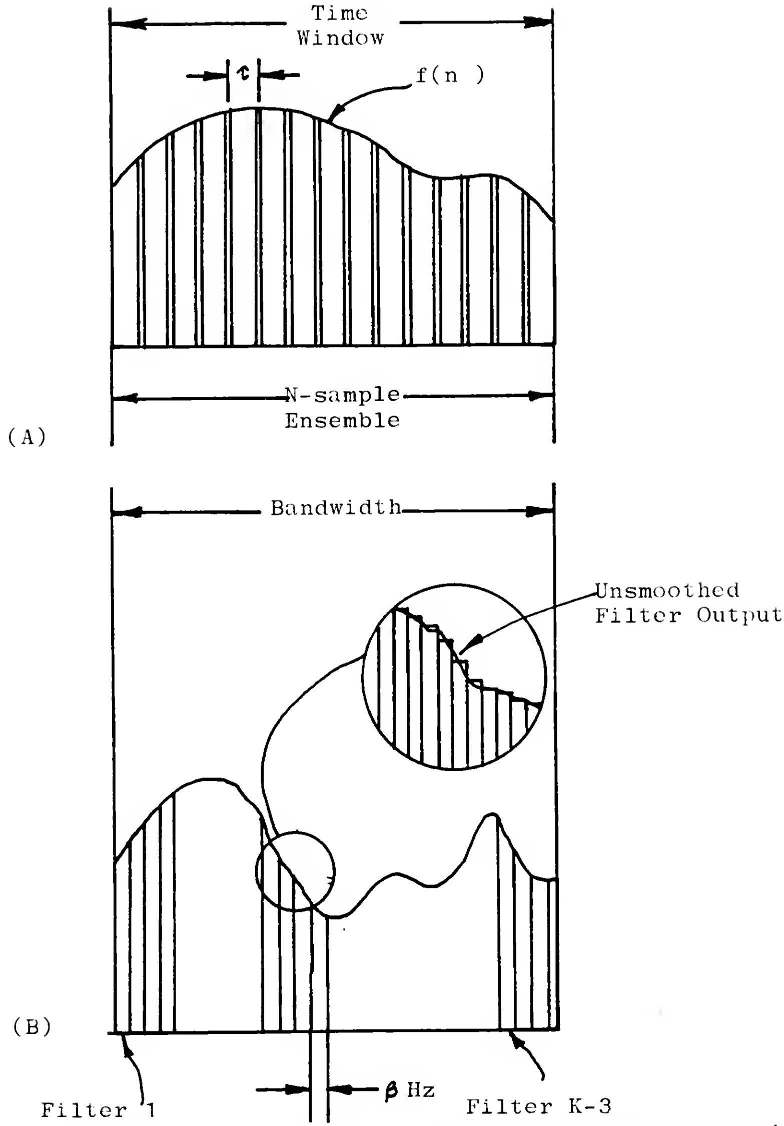


Figure A.1 Conversion of time domain signal, (A) into frequency domain signal, (B) via the fast Fourier transform algorithm. An analog signal of length T is divided into N samples with a sample taken every t seconds. The computation of the discrete Fourier transform from this N-sample ensemble results in K-sample frequency spectrum.

long computation period for application in a real time analyzer.

An enhanced method of computation is made possible by the Fast Fourier Transform (FFT) algorithm developed by Cooley and Tukey (52). They observed that because of the trigonometeric character of

$$e^{-j\frac{2\pi(n)k}{N}},$$

substitution of consecutive integer values of n and k into Equation A.4 produces cyclically repetitive results; thus, many computations need not be made. The FFT is able to reduce the number of computations to about 2% of the number required by the DFT, yet yields identical results.

A.3 Functional Relationships

The linear, time-invariant system to be investigated may be viewed as a box into which the perturbing signal is fed and from which the response signal is led (see Figure A.2). One can now define a family of linear time domain and frequency domain functions which describe the relationship between input and output. The functions appearing in Figure A.2 are defined as follows:

$x(t)$ = time domain expression of perturbing signal

$y(t)$ = time domain expression of response signal

$S_x(f)$ = linear Fourier spectrum of $x(t)$

$S_y(f)$ = linear Fourier spectrum of $y(t)$

$H(f)$ = system transfer function

$h(t)$ = inverse Fourier transform of $H(f)$, sometimes called the system impulse function

Similarly, one can define a family of time and frequency domain functions relating the power of the input signal to the output signal

(see Figure A.3). This set of functional relationships, also known as square law relationships, also provides valuable information about the system. They are defined as follows:

$R_{xx}(\tau)$ = auto correlation of the input signal x(t)

$R_{yy}(\tau)$ = auto correlation of the output signal y(t)

$G_{xx}(f)$ = auto power spectrum of x(t)

$G_{yx}(f)$ = cross power spectrum of y(t) and x(t)

$R_{yx}(f)$ = cross correlation of y(t) and x(t)

Although a detailed explanation of each of these functions and its potential usefulness will not be attempted here, a few more words of explanation are in order.

A.3.1 Auto Correlation

The correlation function is a time average found by multiplying a signal by the same signal lagging τ units in time and averaging the product over the time of observation. It is thus defined as

$$R_{xx}(\tau) = \lim_{T \to \infty} \frac{1}{T} \int x(t) \cdot x(t + \tau) dt \qquad \text{A.5}$$

although it is usually calculated as the inverse Fourier transform of the auto power spectrum. It is most useful in distinguishing a periodic or impulsive signal from random noise (58).

A.3.2 Cross Correlation

The cross correlation measures the similarity between two signals as a function of time shift. It is defined as

$$R_{yx}(\tau) = \lim_{T \to \infty} \frac{1}{T} \int y(t) \cdot x(t + \tau) dt \qquad \text{A.6}$$

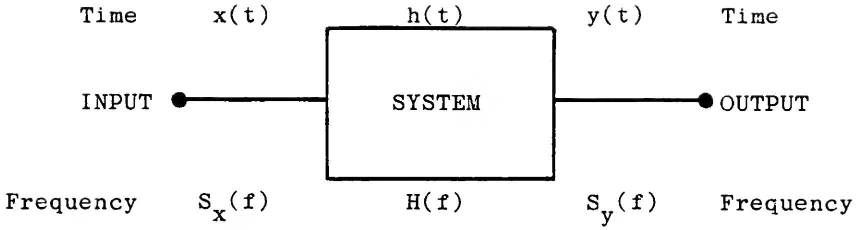


Figure A.2 Linear relationships as defined for digital signal analysis.

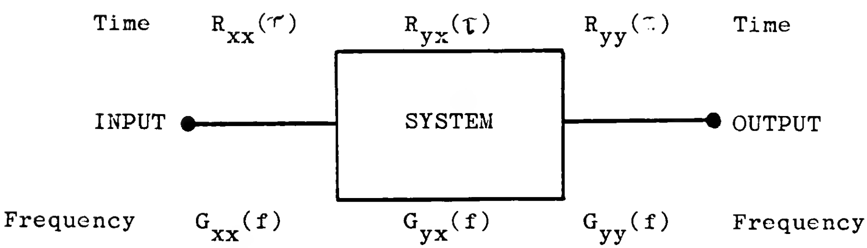


Figure A.3 Square-low relationships as defined for digital signal analysis.

but, just as the auto correlation function, it is usually calculated as the inverse Fourier transform of the cross power spectrum. The cross correlation function is most useful in measuring time delays between signals (58).

A.3.3 Auto Power Spectrum

The auto power spectrum is a function containing only magnitude information about the signal of interest. It is defined as

$$G_{xx}(f) = S_x(f) \cdot S_x(f)^* \qquad \text{A.5}$$

where $S_x(f)^*$ is the complex conjugate of $S_x(f)$. This multiplication effectively removes the imaginary component thus obviating any phase information contained in S_x. As mentioned above, it is also the Fourier transform of the auto correlation function and contains the same information.

A.3.4 Cross Power Spectrum

The cross power spectrum is a measure of the mutual power between two signals at each frequency in the analysis band. Defined as

$$G_{yx} = S_y(f) \cdot S_x(f)^* \qquad \text{A.6}$$

it contains the relative phase between the two signals. It is used to analyze phase relationships between signals.

A.3.5 Transfer Function

The transfer function or frequency response of a system provides magnitude and phase information about the output-input relationships or a system. It is defined as

$$H(f) = \frac{\overline{G_{yx}(f)}}{\overline{G_{xx}(f)}} \qquad \text{A.7}$$

where the bars indicate average values.

Two other functions which can be derived from the computation of those previously listed are the coherence function and the signal-to-noise ratio (S/N). The coherence function assumes values between zero and one and is a measure of the degree of causality between system input and output. It is defined as

$$\gamma^2(f) = \frac{\overline{G_{yx}(f) \cdot G_{yx}(f)^*}}{\overline{G_{xx}(f)} \cdot \overline{G_{yy}(f)}} \qquad 0 \leq \gamma^2 \leq 1 \qquad \text{A.8}$$

A value of $\gamma^2(f) < 1$ at any particular frequency indicates the presence of noise, non-linearities or time delays in the system. The signal-to-noise ratio provides the same information but is easier to grasp intuitively. It is derived from the coherence function by the equation

$$\frac{S(f)}{N(f)} = \frac{\gamma^2(f)}{1 - \gamma^2(f)} \qquad \text{A.9}$$

Another function of the HP5420 signal analyzer is particularly useful in characterizing input signals. It is the amplitude histogram which is a representation of the probability density of the input waveform. As the waveform is sampled, the analyzer counts the number of times a particular voltage is encountered and portrays number versus voltage. Dividing this number by 512 times the number of averages normalizes it to the total number of samples. Using this display, the random noise signal is depicted as a Gaussian distribution.

A.4 Signal Processing Steps

The signal processing steps performed by the signal analyzer in transforming time domain input and output signals to the various

frequency domain functions will now be considered. As mentioned previously and illustrated in Figure A.1, the computation of the DFT or FFT requires that the time domain signal be sampled. This means that the analog voltage signal is viewed for a certain time interval and is a digitized value at each of N equally-spaced subintervals. In the parlance of digital signal analysis, the total time interval in which the analyzer is looking at the input signal is called the time window (T_W), each of the N digitized voltages is called a sample, and the complete set of numbers characterizing the signal during the time window is known as an ensemble. In the Hewlett-Packard 5420 digital signal analyzer, N is 256 for the majority of measurements. Therefore, the required signal sampling rate is determined by the size of the time window, i.e., $256/T_W$.

The mechanism of sampling an analog signal is the electronic multiplication of the time domain signal by the sampling function. The sampling function is a series of unit amplitude pulses occurring at the sampling rate. If one could view the Fourier spectrum of the resultant signal immediately after the multiplication, it would correspond to the spectrum of the original analog signal plus an infinite number of sum and difference frequency components called "aliases". Since the lowest frequency at which an alias can occur is one-half the sampling rate, the sampling rate must be at least twice the highest frequency of interest in the spectrum of the signal of interest. (In the HP5420, it is four times the frequency of interest.) All frequencies from the input to the sampling circuit by a low pass or "anti-aliasing" filter.

The continuous stream of samples coming from the two analog-to-digital converters (ADC) are simultaneously grouped into ensembles of 256 samples each. Using these ensembles of data, the discrete Fourier transform algorithm converts the digitized array into the frequency domain. The FFT computation, although rapid, takes a finite amount of time and begins as soon as there are 256 samples in the ensemble. Figure A.4 is a block diagram illustrating these signal processing steps (59).

A.5 Ensemble Averaging

Signal averaging is used to reduce variance when analyzing random data. The simplest and most familiar form is summation averaging which is performed on a specified number of ensembles. The summation average of N ensembles is given by

$$A_N = \frac{1}{N} \sum_{i=1}^{N} Z_i \qquad \text{A.10}$$

Summation averaging requires that all N ensembles be averaged before a calibrated result is produced and is, therefore, not used in the HP5420 signal analyzer.

Instead, a process called stable averaging is used. Stable averaging produces the same results as summation averaging but maintains its calibration throughout the measurement. The formula for stable averaging is

$$A_N = A_{N-1} + \frac{Z_N - A_N -1}{N} \qquad \text{A.11}$$

Here, A_N is the average after N ensembles. Stable averaging is useful

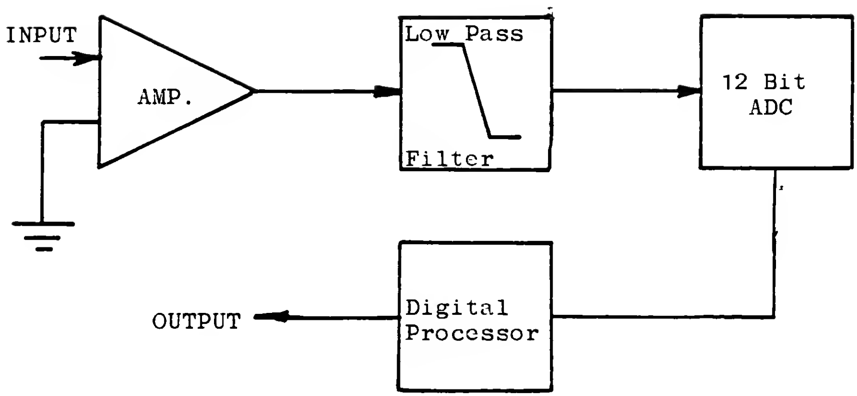


Figure A.4 Schematic illustration of signal processing steps in the HP5420A.

when the characteristics of the system do not change during the averaging process.

In many electrochemical systems, the properties of the system can be expected to change with time and an averaging process is required which will reflect these changes. Such a feature is provided by exponential decay averaging which produces a calibrated average using the last K ensembles. The formula for exponential decay averaging is

$$A_N + A_{N-1} + \frac{Z_N - A_N - 1}{K}$$

where K, known as the decay constant, is a number smaller than N selected by the operator.

This process gives added weight to the latest data and discounts old data more and more. The net effect is to permit observation of permanent changes in the system response with time while averaging out the random fluctuations. While it is envisioned that exponential averaging would be the averaging method of choice while observing a slowly-changing electrochemical system, stable averaging probed to be entirely adequate for the dummy cell and real systems in this work.

APPENDIX B
OPTICAL COUPLER CIRCUITRY

Commercially available optical coupling chips are used primarily in digital switching devices. The operation of the optical coupler is simple in principle. The input signal modulates the intensity of a light-emitting diode. Modulated light traverses a nitrogen-filled, hermetically-sealed container and impinges on a phototransistor which behaves as a conventional transistor with a signal applied to the base-emitter junction. The characteristic response of this coupling method is nonlinear and is sensitive to temperature. However, by using another optical coupler as a feedback element, the nonlinear transfer characteristics can be minimized and the effect of temperature on device characteristics can be reduced.

The topical isolator circuitry, built for the AC-system, is designed to handle floating grounds and has five stages: input amplifier, coupler feedback network, output offset, buffer amplifier and output gain. (See Figure B.1.) The input amplifier stage I1 is a differential amplifier with switched feedback resistors allowing for gains of either 0.1 or 1.0. In this configuration, noise common to both input leads is cancelled out. Signal attentuation, if required, prevents operation of the optical couplers in a nonlinear fashion.

The network consisting of I2, O1 and part of O2 sets the bias voltage for the optical couplers and provides negative feedback to enhance the linearity of the output signal. The bias adjust

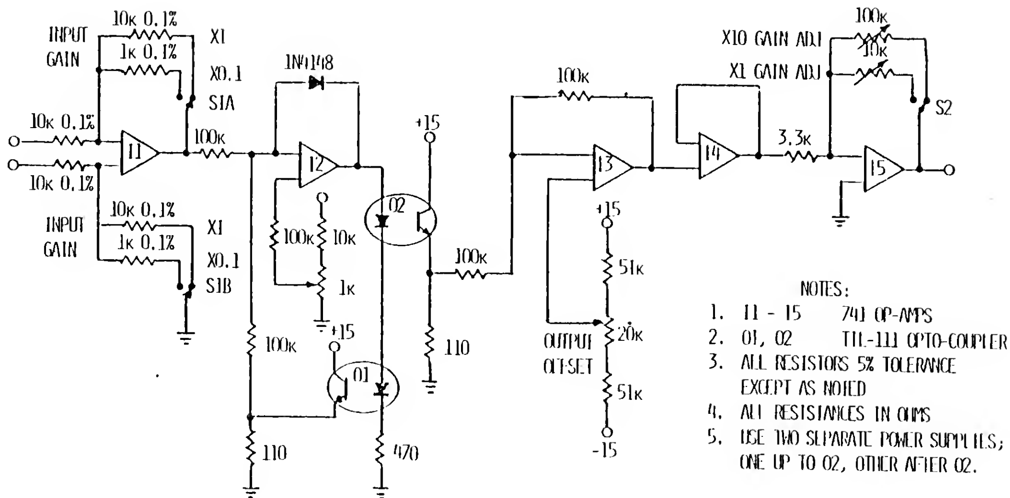


Figure B.1 Schematic diagram of the optical isolation circuit.

potentiometer in conjunction with I2 is used to set the most linear operating point for O1 and O2. Changes in gain due to noise or temperature changes are sensed by O1 and fed to the inverting input of I2 to stabilize the gain.

The output offset potentiometer knob is located on the exterior of the isolator circuit chassis permitting adjustment of the DC value of the output signal to match that of the instrument connected to the output of I5. This stage is controlled by I3 which sums the signal from O2 and the output offset potential from the potentiometer. I4 is a buffer amplifier with high input impedance, low output impedance and a gain of 1. The output gain stage consisting of I5 and switched trimming potentiometers permits calibration of the entire isolator circuit to yield overall gains of 1.0 and 10.0.

Without separate power supplies and grounds for input and output stages, the circuit would not provide isolation. Accordingly, one is obliged to power circuit components up to O2 with one $\pm$ 15 volt power supply and those after O2 with another $\pm$ 15 volt supply. All circuitry up to and including pins 1 and 2 of O2 belong to the input stage and all circuitry from and including pins 4 and 5 of O2 is output circuitry.

As shown in Figure B.2, the optical isolation circuit has a reasonable flat frequency response up to about 15 kHz. This curve was determined by measuring the transfer function of the isolation circuit with the HP5420. Beyond 15 kHz, the phase lag becomes increasingly significant. The large amplitude spike approximately midway across the plot and the severed smaller spikes immediately to its right are caused by 60 Hz noise and its harmonics, respectively.

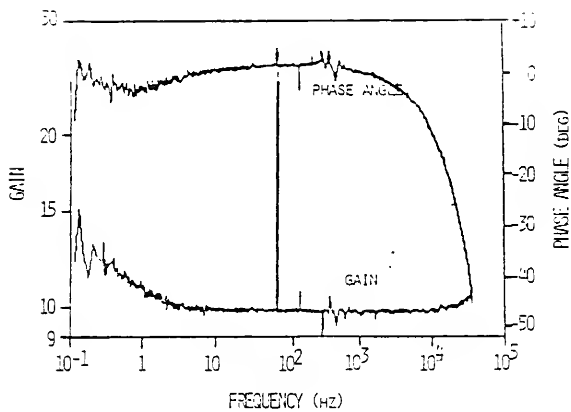


Figure B.2 Frequency response of the optical isolation circuit.

APPENDIX C
SIMULATED ELECTRODE STUDIES

In this sequence of experiments on a single simulated electrode, the effects of signal amplitude, potentiostat incorporation and the use of optical isolation were assessed. All experiments were performed on a three-components electronic network with $R_s = 20.0 \pm 0.4\ \Omega$, $R_p = 1000 \pm 1\Omega$ and $C_D = 22 \pm 2\mu F$.

C.1 Effect of Signal Amplitude on Data Quality

The response of a simulated electrode network was measured for signal amplitudes of 4v, .4v and 120 mv p-p with the BLWN signal applied across the network and current determined by the voltage drop across R_s. See Figure C.1 for the wiring of this experiment series. The results are given in the plots of Figures C.2-C.4, respectively. Note that the data are portrayed in the complex plane, Pohlman-Smyrl and Bode formats. The complex plane plot appears "upside-down" because + imaginary rather than - imaginary values have been plotted.

The data are superimposed on theoretical curves determined using the nominal vlues of network components. It can be seen in Figure C.2 that the experimentally determined values of R_p and R_s are very close to their nominal values. The disparity in slopes evidenced in Figure C.2g reflects a deviation in C_D from the nominal value; however, analysis shown the deviation to be less than 5%, well within the allowable tolerance for the capacitor. Data quality is observed to deteriorate at low frequencies for the reasons discussed in Chapter III.

Lower signal amplitude does not necessarily mean deterioration in S/N as shown in Figure C.3. There is no noticeable change in data quality as a consequence of the ten-fold decrease in signal amplitude. In Figure C.4, however, there is a marked increase in low frequency data scatter, a trend which worsened with further decreases in signal amplitude. Plot character was found to be virtually unrecognizable in the 20 mv p-p range. For this particular network, where the voltage drop across R_s is about 2% of the voltage drop across the entire network, a 20 mv p-p perturbation would result in a 0.4 mv p-p voltage drop across R_s. Given the capabilities of a 12 bit ADC, such a signal can be resolved into only 8 discrete levels.

The graphical portrayals in Figures C.2-C.4 were analyzed for the values of R_s, R_p and C_D which they would predict. The comparison of predicted and nominal values for the three signal amplitudes as shown in Table C-1. As can be seen, one of the three analysis techniques was capable of predicting all values to within $\pm 5\%$ of the nominals for each of the three amplitudes.

C.2 Simulated Electrode Under Potentiostatic Control

To demonstrate that data could also be obtained with an electrode under potentiostatic control, the network was connected between the counter and working electrode leads of the potentiostat and a 160 mv p-p BLWN signal was applied to the summing junction of the control amplifier. (See Figure C.5 for a schematic representation.) Potential drop across the network was monitored with the potentiostat electrometer and current flow measured as the voltage drop across R_s .

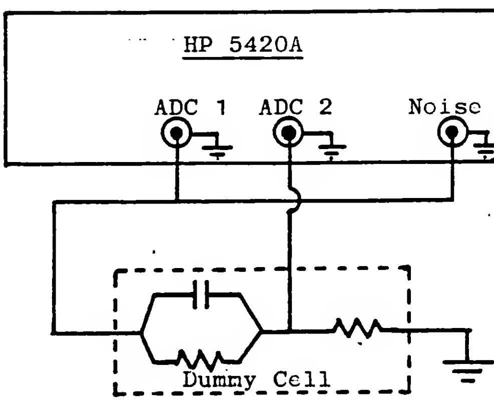


Figure C.1 Schematic illustration of connections without optical isolation or potentiostat.

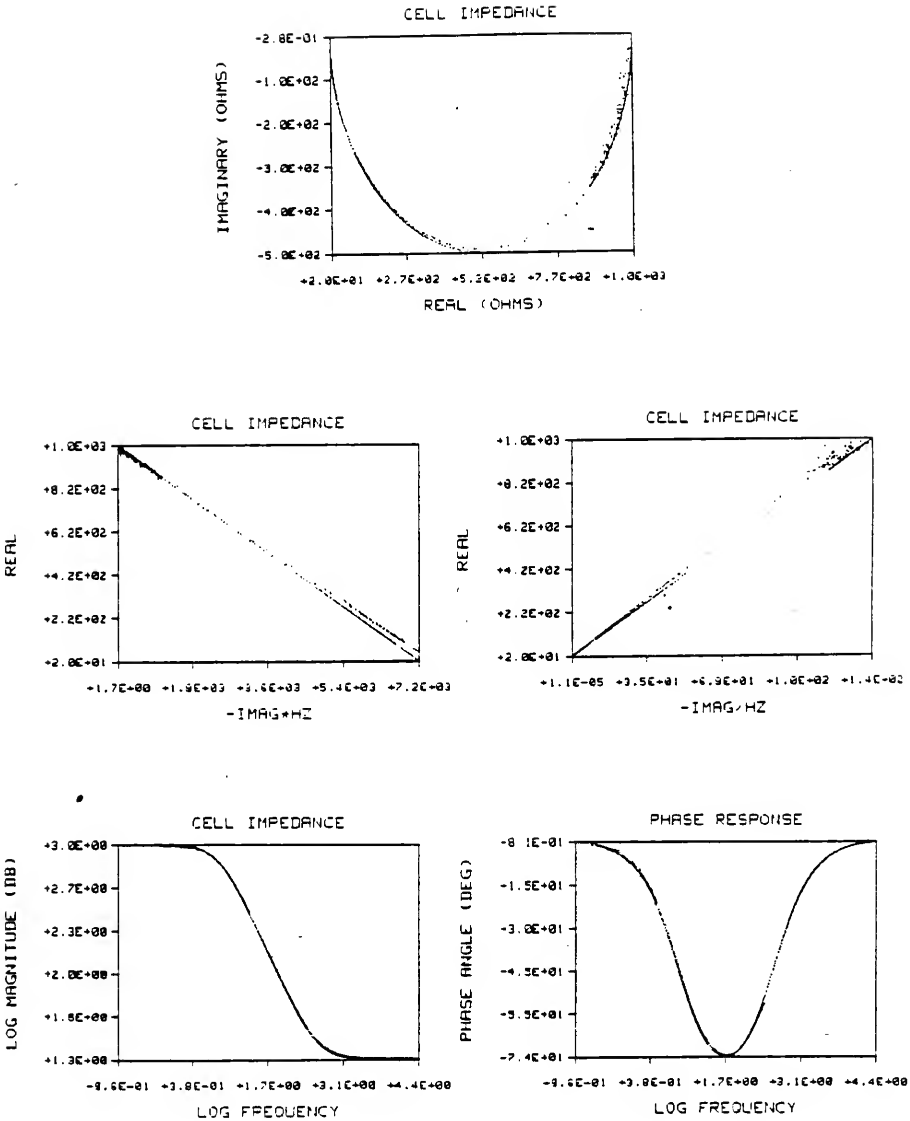


Figure C.2 Three-element network impedance plots for signal amplitude of 4.0 volts.

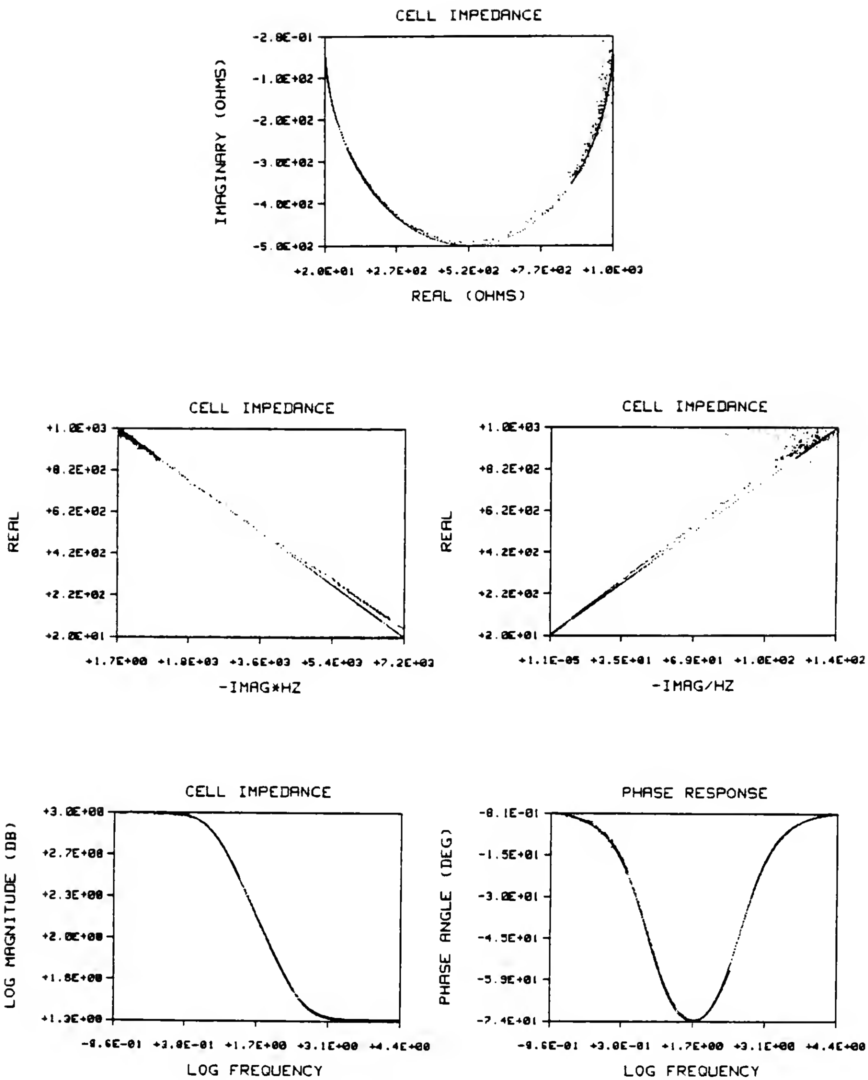


Figure C.3 Three-element network impedance plots for signal amplitude of 0.4 volts.

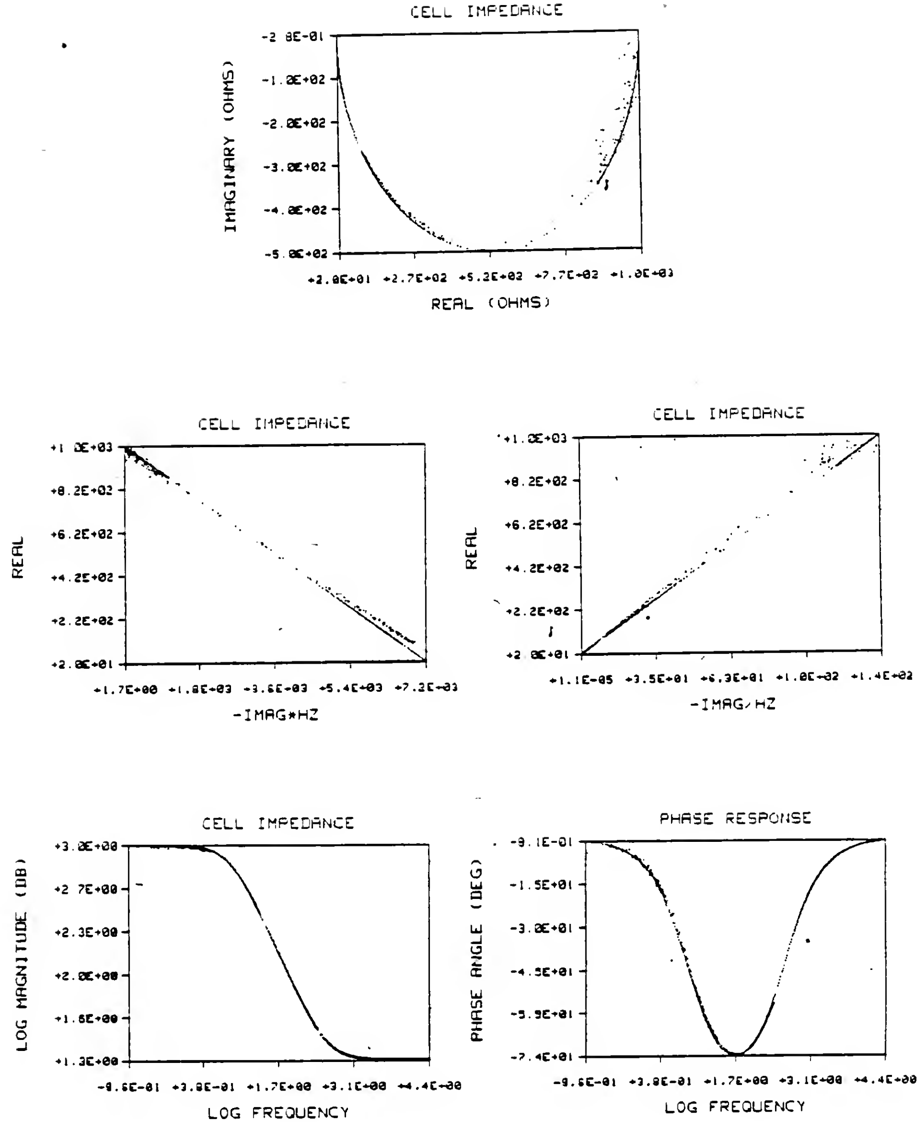


Figure C.4 Three-element network impedance plots for signal amplitude of 120 mv.

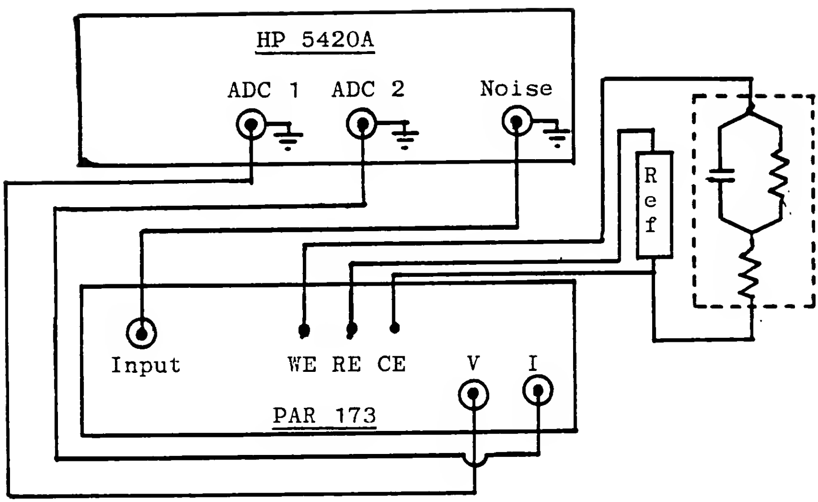


Figure C.5 Schematic illustration of connections with a potentiostat.

Table C-1 Comparison of Analytical Findings As A Function of Signal Amplitude

		Analysis Type					
		Pe vs Im		Re vs Im *Hz and Re vs Im/Hz		Bode Analysis	
R_p (Ω)	nom	1000	% Δ	1000	% Δ	1000	% Δ
	4v	997.56	−0.24	994.01	−0.60	990.40	−0.96
	.4v	999.86	−0.01	988.00	−1.20	979.99	−2.00
	.12v	1021.96	+2.20	978.78	−2.12	983.69	−1.63
R_s (Ω)	nom	20	% Δ	20	% Δ	20	% Δ
	4v	20.33	+1.65	21.06	+5.30	20.33	+1.65
	.4v	20.33	+1.65	20.92	+4.60	20.33	+1.65
	.12v	20.33	+1.65	21.61	+8.05	20.33	+1.65
C_D (Ω)	nom	22	% Δ	22	% Δ	22	% Δ
	4v	23.01	+4.59	21.17	−3.77	21.38	−2.82
	.4v	22.33	+1.50	21.28	−3.27	22.34	+1.55
	.12v	24.92	+13.27	21.34	−3.00	21.45	−2.50

These connections resulted in a DC offset of about 20 mv at the HP5420 ADC, and an offset of 20-30 mv at the potentiostat. The presence of such offsets led to the development of optical isolation discussed in Chapter 3 and Appendix B; however, the offsets were ignored during execution of this run. As can be seen in the sequence of plots of Figure C.6, the data have the same character as that obtained without potentiostatic control but exhibits more data scatter, an expected outcome when the signal is processed through additional analog amplifier stages.

C.3 Effect of Optical Isolation

Figure C.7 is a series of plots made with the same signal amplitude and potentiostatic control but with the input signal separated from the potentiostat with an optical isolator. The inputs to both HP5420 ADC channels were also optically isolated as shown in Figure C.8. The offset potentiometers of the isolators were used to eliminate all DC offsets. Although the plot character is maintained, there is more data scatter evident in Figure C.7 than exhibited by the network when subjected to a 120 mv p-p signal without potentiostatic control (Figire C.4). The price one pays for elimination of DC offsets is a deterioration of S/N.

Another penalty observed in the plots of Figure C.7: the frequency response above 10 kHz exhibits instrumental artefact from the isolators, destroying the usefulness of the experimental data in this range of frequency. One could attempt to correct this problem by using optical isolator chips with better high frequency response characteristics or could simply be content with the low frequency data.

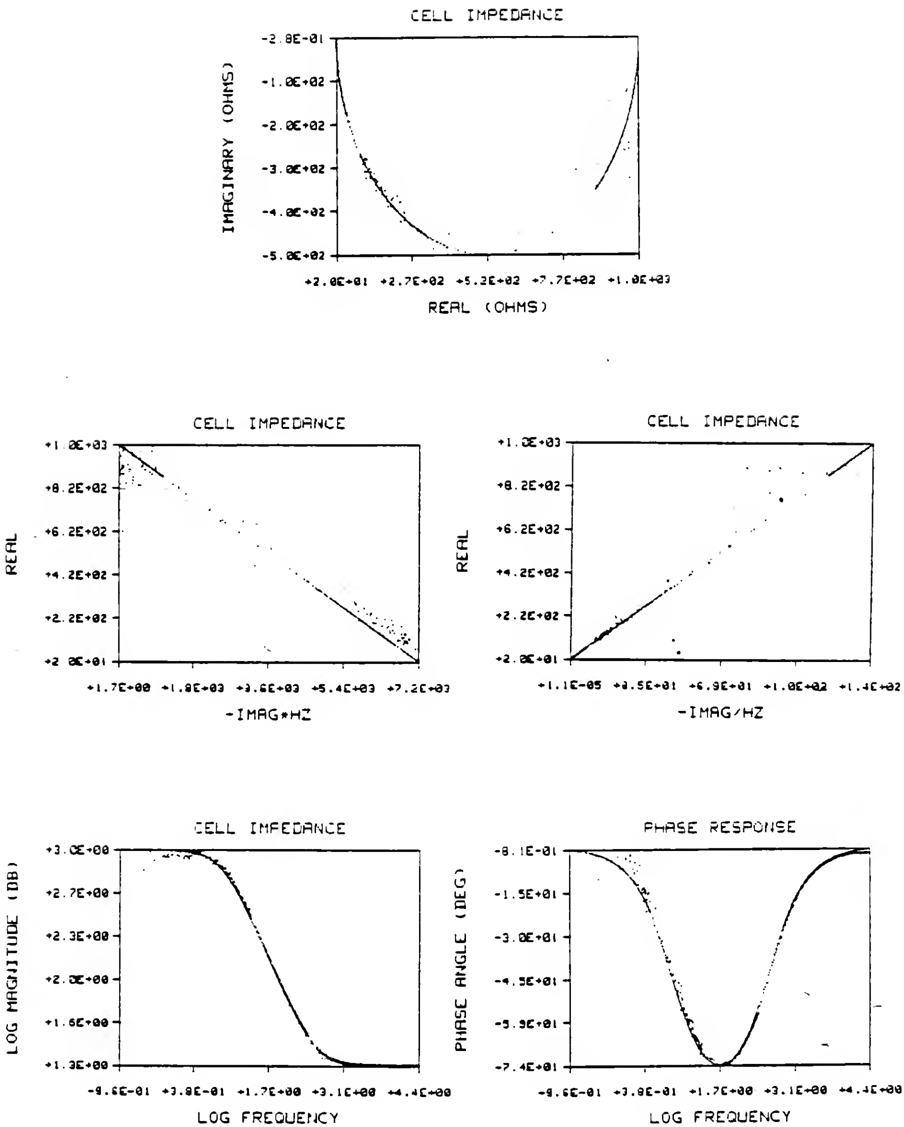


Figure C.6 Three-element network impedance plots made with a potentiostat in place.

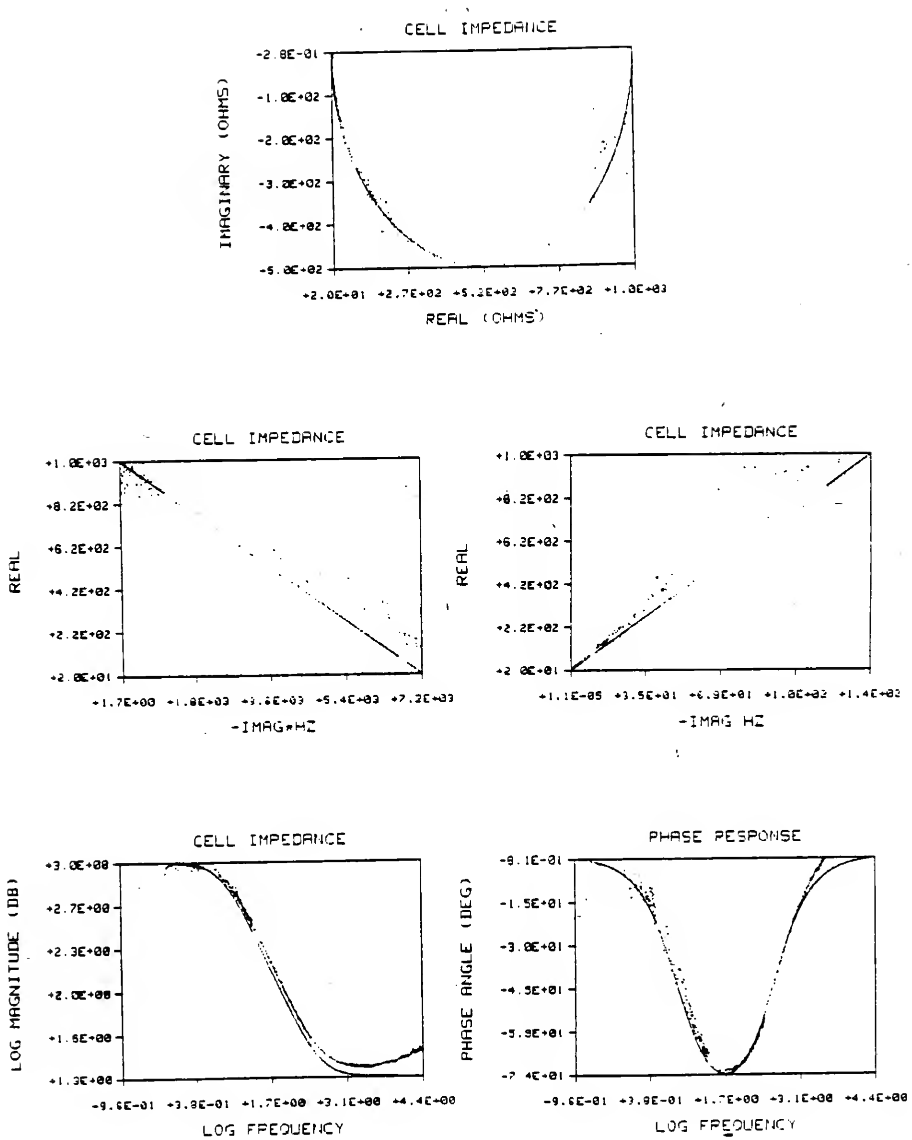


Figure C.7 Three-element network impedance plots for measurements made with potentiostat and optical isolators in place.

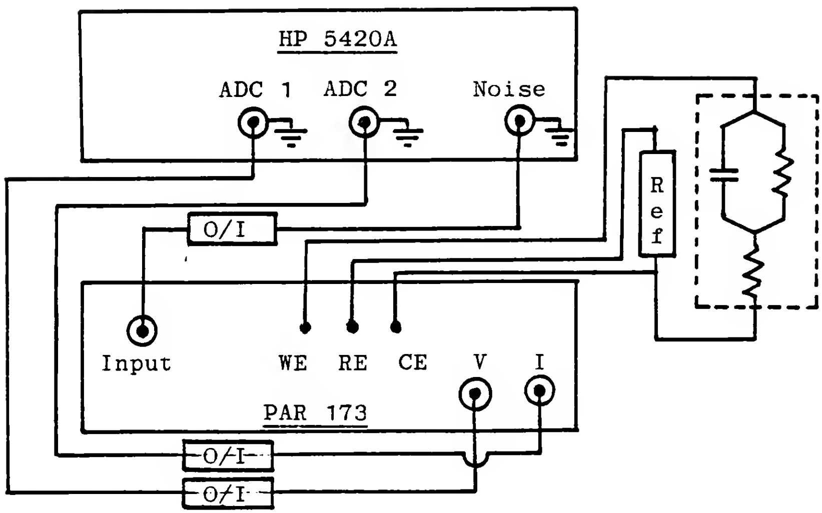


Figure C.8 Schematic illustration of connections to a three-element network with potentiostat and optical isolation.

APPENDIX D
SYSTEM SOFTWARE

The computer software created during the course of this project falls into three general categories: (1) the BASIC language main program written by Bob Fortune to store, manipulate, smooth, fit, analyze and plot data gathered by the HP5420 signal analyzer; (2) a series of command files written by the author using the main program to perform routine data gathering and analysis procedures; and (3) a BASIC language program, GRAFIT, for use with a HP85 microcomputer and HP7225 Graphics Plotter in plotting analytical portrayals of impedance for the five-element network models described in Chapter IV.

The Main Program

The main program, designated at 5420N1, may be read into RAM from either tape or eight-inch diskette. However, the diskette also contains subroutines which are called by the main program during the course of execution. The diskette must, therefore, be in place in an operational disk drive during execution of the main program. The range capabilities of the program is illustrated by the menu which is displayed whenever this program is run.

5420 ANALYSIS UNIT GROUP MENU

K0 = SETUP
K1 = CURSOR
K2 = DISPLAY
K3 = CONTROL
K4 = SPECIAL
K5 = MODE
K6 = COMMAND FILE FUNCTIONS
K7 = DATA FILE FUNCTIONS
K8 = DATA MANIPULATIONS
K9 = EDIT FUNCTIONS
K10 = DELETE LOCAL COMMANDS
K11 = FILE-OF-FILES
K12 = COMPUTER OPERATIONS
K14 = PLOT LIMITS
ΛK4 = COMMAND DUMP

In this menu listing, the letter K followed by a number refers to a function key which may be depressed to activate the indicated section of software. The function and capabilities of each of the individual sections will now be outlined.

Function Keys

K0 SET UP. When function key 0 is activated, subroutine "Set up_group" is read in from the diskette and executed. The purpose of the setup group is to permit the user to operate the setup buttons in the 5420 analyzer with computer control. The user creates a series of command which upon execution are transmitted via the HPIB to the HP5420 and place it in a configuration capable of receiving data. For example, one can specify whether the analyzer is to be configured for frequency or time domain analysis, which type of averaging is desired, the type of input signal which will be used, etc. Once the user understands how to operate the keys manually, the use of the set-up group program is self-explanatory.

K1 CURSOR. When function key 1 is activated, subroutine "Cursor_group" is read in from the diskette and executed. The cursor group was to have permitted control of the cursor on the CRT display of the 5420 analyzer. This group was never fully developed and is of no use.

K2 DISPLAY. When function key 2 is activated, subroutine "Display_group" is read in from the diskette and executed. The purpose of the display group is to create commands which specify in which plot format a given set of data is to be displayed on the CRT of the HP5420. For example, the data may be portrayed in complex plane or Bode plot formats. All the keys on the HP5420 which control display may be controlled from the HP9845 computer. Use of this group is straightforward once the user understands the function of the HP5420 keys.

K3 CONTROL. When function key 3 is activated, subroutine "Control_group" is read in from the diskette and executed. The control group permits creation of computer commands for the control keys of the HP5420 analyzer. For example, once all setup and display format information has been entered, data collection may begin. This requires activation of the START control. Other control keys are PAUSE/CONTINUE, VIEW INPUT, MAX RATE, RESET and SELF-TEST.

K4 SPECIAL. When function key 4 is activated, subroutine "Special_group" is read in from the diskette and executed. The special group contains miscellaneous commands which can be used with the HP5420. For example, commands to prohibit recognition of local (manual button-pushing) commands during computer-controlled command execution is provided here. One can also place text on the screen of the 5420 CRT using this group.

K5 MODE. When function key 5 is activated, subroutine "Mode" at line 1320 of the main program is executed. Mode control determines whether a set of commands will be executed immediately, stored for later execution, or whether a stored set should be executed. This group is used to execute command files.

K6 COMMAND FILE FUNCTIONS. When function key 6 is activated, subroutine "Command_file" is read in from the diskette and executed. This group permits manipulation of command files after creation. One may save, recall, or append an existing command file.

K7 DATA FILE FUNCTIONS. When function key 7 is activated, subroutine "Dfile" is read in from the diskette and executed. Data files are created each time a set of data is gathered by the HP5420. The data file function group permits these so-called short data sets to be named and renamed, tied together into a long-data set (deleting low-resolution over-lapping data points) and resaved. This group creates instructions which are included in a command file to perform these functions routinely.

K8 DATA MANIPULATIONS: When function key 8 is activated, subroutine "Dmanip" is read in from the diskette and executed. Once data files have been created and stored, this group permits mathematical manipulation of the data. For example, individual short data sets can be tied together. Lease squares and polynomial fits may be performed on the data, generating new smoother sets of data. Real and imaginary arrays may be scaled by a factor. Using the cursor, regions of interest may be specified, creating new data sets for the region of interest. This group proved to be an extremely flexible and useful tool.

K9 EDIT FUNCTIONS. When function key 9 is activated, subroutine "Edit1" at line 1590 of the main program is executed. This group constitutes an editing package for the command files. The user may delete or replace individual lines or list the entire command file. In using the editor, other groups such as SETUP, CURSOR, DISPLAY, CONTROL, DATA FILES, and DATA MANIPULATION may be invoked.

K10 DELETE LOCAL COMMANDS. When function key 10 is activated, subroutine "Del_com" at line 2550 of the main program is executed. This subroutine deletes all commands currently in the local command file, thus preparing for the creation of a new command file.

K11 FILE-OF-FILES. When function key 11 is activated, subroutine "Mul_file" is read in from the diskette and executed. This subroutine provides for the sequential execution of a series of command files. A particular sequence such as data collection, followed by plotting in a particular format, can be given a file-of-files name, which can then be executed at will.

K12 COMPUTER OPERATIONS. When function key 12 is activated, subroutine "Com_group" is read in from the diskette and executed. This group should be more logically named "PLOTTING GROUP" as it is concerned with defining the type of plot portrayal for the data, X and Y axis labeling, and whether the plot will be placed on the CRT, printer, or remote four-color plotter.

K14 PLOT LIMITS. When function key 14 is activated, subroutine "Plot_group" is read in from the diskette and executed. The purpose of this group is to establish the plot limits, location and scale and to identify whether the plot is new or will be an overlay on an old plot.

ΛK4 COMMAND DUMP. This key, which means "shift key 4", activates function key 20 which causes subroutine "C_dump" at line 7425 of the main program to be executed. Depressing the shift and key 4 simultaneously causes the series of commands in the local command file to be listed by the integral thermal printer for inspection.

Main Program Listing

The main program consists of numerous subroutines which are executed in an order dependent on the desires of the user. Table D-1 summarizes the functions of the various subroutines which make up the main program. A complete listing of the main program follows Table D-1.

TABLE D-1 MAIN PROGRAM SUBROUTINES

LINE #	NAME	FUNCTION
525, 560, 595, 630 670, 1500 3190, 3230 4725, 5730 6835	SUB KEY_N	When function key N is pressed, a subroutine stored on the diskette is transferred to a particular RAM location and executed.
425	Main Menu	Displays the 5420 Unit Group Menu on the CRT.
355	Clear	Clears the top 20 lines of the CRT display.
385	Putline	Displays a line in a specified format on the CRT.
815	Send	Sends a series of commands in a command file to HP5420
1185	Shorty	Shortens a tied-data set spanning 5 decades of frequency to 183 data points equally spaced over the log frequency range.
1320	Mode	Stored in RAM, called by depressing Key 5. Allows user to select whether a command file in RAM should be executed immediately or stored; or if stored, command file should be executed.
1590	EditL	Subroutine which permits editing of a command file.
2165	Isr	Identifies status codes of HP5420 during execution of a command file and tells computer what to do about them.
2375	Make_plot	Sends data on the interface bus between HP5420 and HP9845 depending on status code identified in Isr.
2550	Del_com	Stored in RAM, called by depressing key 10; deletes all commands from local command file.
2705	Savrec	Saves or recalls data for single bandwidth data set collected by HP5420.

Table D-1--continued.

LINE #	NAME	FUNCTION
3085	Adc_ovrflo	Beeps when Analog to Digital converter encounters an overflow condition during a run.
3130	C_err	Converts HP5420 status codes into error identification.
3285	Clear_line	Clears a line of the command file.
3310	Get_p	A subroutine used by the keys which permits HP5420 command set-up. Prompts for the command information.
3435	Shifty	Renumbers command file when command deleted or added
3565	Group_menu	Function unknown.
3660	Label	Label a plot format.
3770	Sys	System codes are discussed in more detail later. This subroutine contains the logic introduced into the command file through the use of a particular system code.
4155	Movr	Renames real and imaginary data points to a form which can be plotted.
4265	Data_file	Stores and recalls data sets collected by HP5420.
4765	Tie	Ties consecutively gathered data sets together discarding lot resolution overlapping data, creating a tied data set.
5020	Linearc	Generates plot parameters, labels and draws axes for a plot.
5775	Datam	Subroutine to manipulate data sets as directed by system codes. For example, swaps real and imaginary arrays or scales real and or imaginary by a constant.

Table D-1--continued.

LINE #	NAME	FUNCTION
6045	Abuild	Builds appropriate arrays of data for plotting all desired formats.
6670	Plfix	Locates the position and determines the scale of the desired plot.
6880	Plot_fix	Uses system code information to obtain plotting information for Plfix.
7210	Smoothr	Contains logic based on system codes for determining how a data set should be smoothed.
7320	Smooth	Performs a data smoothing algorithm on a data array.
7425	C_dmp	Activated by shift key 4; dumps contents of the local command file to the thermal printer of the HP9845.
7485	S plot	Contains plotting logic based on system code information.
7705	Zero_data_sets	Assigns all values of a named data file to zero.
7860	Ren_file	Permits a command file to be renamed.
7975	Del_files	Permits a command file to be deleted.
8080	L1	Function unknown.
8205	Points	Permits the cursor to be moved along a plotted data array to identify coordinates of specified points. Enabled with system code 41.
8560	Pointr	Function unknown.
8600	Least	Performs a least squares analysis on data.
9330	Fit	Allows user to select a region of data over which a least squares fit is performed.

Table D-1--continued.

LINE #	NAME	FUNCTION
9865	Swap_scale	Function unknown.
10050	Unswap_scale	Function unknown.
10175	Antol	A complicated method for averaging data.
10565	Lsfit	Performs a linear least squares fit on data, plots it and determines equation of line.
11315	Lsauto	Function unknown.
11585	Manul	Function unknown.
12030	Region_intrest	Determines a region of interest over which data may be evaluated by a fitting technique.
12420	Analysis	Performs graphical analysis on five types of plots, fitting data to three-element network model.
13045	Remove	Deletes data points in the vicity of 60 Hz and its harmonics.
13420	Command_file	Permits saving, appendings and recalling command file stacks by name.

Main Program Listing

```
1     !                                                             !
2     !                                                             !
3     !                                                             !
4     !                                                             !
5  OPTION BASE 1
10    COM Command$(200),Number_commands,Mode$,Menu,A5420,Word$(200),Prefix$(200)
15    COM Editf$,Edit_line,B5420,SHORT Pdata(1),Nprint,Sdata(528),Nsave,Stat
20    COM File$,SHORT State(1),Ry(512),Ix(512),Sz(30),Xline,Tc,Tf,Ti,Np,Slope,P1
,P2
25    COM SHORT Xstart,Xstep,Ryc(1536),Ixc(1536),Frc(1536),Cp,Intercept,Frequenc
y
30    COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax
,Paxis
35    COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4.D5,D6,D7,
D8,D9
40    COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
45    COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(128),Ty(128),Tflg,X60,E
very
50    COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$
[20]
55    ! COM SHORT CPX,XTP(1536),CPC,CTP(1536)
60    D9=-1
65    Region=0
70    Paxis=0
75    Sflg=0
80    Pflg=0
85    Lsf=0
90    Editf$=" "
95    B5420=0
100   Pscale=15
105   Number_commands=0
110   Onn=1
115   Off=1
120   Blue=0
125   Black=1
130   Pager=1
135   Nopager=0
140   Every=0
145   X60=1
150   Frequency=Every
155 A5420=704 !  ADDRESS OF 5420 UNIT
160 ON INT #7,12 CALL Isr
165 CONTROL MASK 7;128
170 CARD ENABLE 7
175 PRINTER IS 16
180 ASSIGN #6 TO "5420LF:F9"
185 File$="5420LF:F9"
190 READ #6;Number_commands
195 MAT READ #6;Command$,Word$,Prefix$
200 ASSIGN * TO #6
205 ON KEY #15 GOTO 285
210 ON KEY #0 CALL Key_0 !Setup_group
215 ON KEY #1 CALL Key_1 !Cursor_group
220 ON KEY #2 CALL Key_2 !Display_group
225 ON KEY #3 CALL Key_3 !Control_group
230 ON KEY #4 CALL Key_4 !Special_group
235 ON KEY #5 CALL Mode
240 ON KEY #6 CALL Key_6 !Command_file
245 ON KEY #9 CALL Editl
250 ON KEY #10 CALL Del_com
255 ON KEY #11 CALL Key_11 !Mul_file
260 ON KEY #12 CALL Key_12 !COM_group
265 ON KEY #7 CALL Key_7   !Dfile
270 ON KEY #8 CALL Key_8   !Dmanip
275 ON KEY #14 CALL Key_14 !Plot_group
280   ON KEY #20 CALL C_dmp
```

Main Program Listing--continued.

```
285 CALL Mainmenu
290 Menu=15
295 IF Menu<>15 THEN 285
300 GOTO 295
305 STOP
310 !
315 !
320 !
325 !
330 ! SUBROUTINES USED IN THIS PROGRAM
335 !
340 !
345 !
350 !
355 !
360 ! SUBROUTINE TO CLEAR THE TOP 20 LINES OF DISPLAY
365 !
370  SUB Clear
375  PRINT CHR$(12)
380  SUBEND
385  !
390  ! SUBROUTINE TO MOVE CURSOR ABOUT SCREEN
395  SUB Putline(Ir,Ic,A$)
400  PRINT USING "#,3A,2D,A,2D,A,K";CHR$(27)&"&a",Ir,"r",Ic,"C",A$
405  SUBEND
410  !
415  ! SUBROUTINE TO DISPLAY MAIN MENU
420  !
425  SUB Mainmenu
430  CALL Clear
435  CALL Putline(0,21,"5420 ANALYSIS UNIT GROUP MENU")
440  CALL Putline(2,27,"K0  =  SETUP")
445  CALL Putline(3,27,"K1  =  CURSOR")
450  CALL Putline(4,27,"K2  =  DISPLAY")
455  CALL Putline(5,27,"K3  =  CONTROL")
460  CALL Putline(6,27,"K4  =  SPECIAL")
465  CALL Putline(7,27,"K5  =  MODE")
470  CALL Putline(8,27,"K6  =  COMMAND FILE FUNCTIONS")
475 CALL Putline(9,27,"K7  =  DATA FILE FUNCTIONS")
480 CALL Putline(10,27,"K8  =  DATA MANIPULATIONS")
485 CALL Putline(11,27,"K9  =  EDIT FUNCTIONS")
490 CALL Putline(12,27,"K10 =  DELETE LOCAL COMMANDS")
495 CALL Putline(13,27,"K11 =  FILE-OF-FILES")
500 CALL Putline(14,27,"K12 =  COMPUTER OPERATIONS")
505  CALL Putline(15,27,"K14 =  PLOT LIMITS")
510   CALL Putline(16,26,"^K4  =  COMMAND DUMP")
515  CALL Putline(18,24,"STRIKE DESIRED KEY")
520  SUBEND
525  !
530  !
535  SUB Key_0
540   OPTION BASE 1
545  LINK "KEY_0:F9",13420
550   CALL Setup_group
555  SUBEND
560  !
565  !
570  SUB Key_1
575   OPTION BASE 1
580   LINK "KEY_1:F9",13420
585   CALL Cursor_group
590   SUBEND
595  !
600  !
605  SUB Key_2
610   OPTION BASE 1
615   LINK "KEY_2:F9",13420
```

Main Program Listing--continued.

```
620  CALL Display_group
625  SUBEND
630  !
635  !
640  !
645  SUB Key_3
650  OPTION BASE 1
655  LINK "KEY_3:F9",13420
660  CALL Control_group
665  SUBEND
670  !
675  !
680  !
685  !
690  !
695  SUB Key_4
700 OPTION BASE 1
705  COM Command$(*),Number_commands,Mode$,Menu,A5420,Words$(*),Prefix$(*)
710  COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
715  COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
720  COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
725  COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax
,Paxis
730  COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,
D8,D9
735  COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
740  COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
745  COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Times
[20]
750  IF D9<>4 THEN LINK "KEY_4:F9",13420
755  CALL Special_group
760  D9=4
765  SUBEND
770  !
775  !
780  !
785  !
790  ! SUBROUTINE TO SEND COMMANDS TO 5420
795  ! CALL:     CALL SEND
800  !
805  !
810  !
815  SUB Send
820  OPTION BASE 1
825 COM Command$(*),Number_commands,Mode$,Menu,A5420,Words$(*),Prefix$(*)
830 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
835 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
840 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
845 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,P
axis
850 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D8
,D9
855 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
860 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
865 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Times$[2
0]
870 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
875  DIM L$[25]
880  ON KEY #16 GOTO Bye
885 Lower=1
890 Upper=Number_commands
895 ! DETERMINE UPPER BOUND BASED ON MODE
900  IF Mode$="IMM" THEN Lower=Number_commands  !  IMMEDIATE MODE
905  IF Mode$="EXE" THEN Upper=Number_commands ! SEND STORED COMMANDS
910  IF Mode$="EXE" THEN CALL Putline(15,25,"COMMANDS BEING ACTED UPON")
915  !
920  ! LOOP TO SEND COMMANDS TO 5420
```

Main Program Listing--continued.

```
925  !
930  Length=0
935   FOR J=Lower TO Upper
940  Xline=J
945  Command$(J)=TRIM$(Command$(J))
950  Length=Length+LEN(Command$(J))
955  IF Tc=0 THEN 975
960  PRINTER IS 0
965  PRINT J;Command$(J)
970  PRINTER IS 16
975   IF Command$(J)[1,2]="//" THEN GOTO 1150
980   L$=""
985    L$=CHR$(129)&"  "
990   L$=L$&VAL$(J)
995   L$=L$&"  -  "
1000  L$=L$&Command$(J)
1005  IF Command$(J)="*SYS*" THEN L$=L$&"  "&Prefix$(J)
1010  Lx=LEN(TRIM$(L$))
1015   FOR J1x=Lx+1 TO 23
1020  L$=L$&" "
1025  NEXT J1x
1030  L$[24,25]=CHR$(128)&CHR$(128)
1035  CALL Putline(20,0,L$)
1040 IF Command$(J)="*SYS*" THEN GOTO 1060
1045 IF Prefix$(J)<>"3325A" THEN OUTPUT A5420 USING "K";Command$(J)
1050 IF Prefix$(J)="3325A" THEN OUTPUT 717 USING "K";Command$(J)&"*"
1055 GOTO 1080
1060 Code=VAL(Prefix$(J))
1065 CALL Sys(Code)
1070 J=Xline
1075  GOTO 1150
1080 IF POS(Command$(J),"ST;") AND (LEN(Command$(J))=3) THEN B5420=1   ! START CO
MMAND
1085 IF POS(Command$(J),"PL;") AND (LEN(Command$(J))=3) THEN B5420=1
1090 IF POS(Command$(J),"PL;")=0 THEN 1125
1095 Icomma=0
1100 X$=Command$(J)
1105 FOR K=1 TO LEN(Command$(J))-1
1110 IF X$[K,K]="," THEN Icomma=Icomma+1
1115 NEXT K
1120 IF Icomma=1 THEN B5420=1
1125 IF POS(Command$(J),"SA;") THEN B5420=1   ! SAVE TO TAPE/CONTROLLER
1130 IF POS(Command$(J),"RA;") THEN B5420=1   ! RECALL FROM TAPE/CONTROLLER
1135 IF POS(Command$(J),"RS;") THEN WAIT 5000
1140 ! IF POS(Command$(J),"GL;") AND (LEN(Command$(J))=3) THEN B5420=1    ! LOCA
L
1145 IF B5420 THEN 1145
1150  NEXT J
1155 ! ONCE YOU HAVE EXECUTED COMMANDS, SET MODE TO STORE
1160  IF Mode$="EXE" THEN Mode$="STO"
1165 Bye:  !
1170  CALL Putline(20,5,"                          "&CHR$(128))
1175  Menu=-100
1180  SUBEND
1185  !
1190  !
1195  !
1200  !
1205  SUB Shorty(SHORT Cp,Ryc(*),Ixc(*),Frc(*))
1210  OPTION BASE 1
1215  Oldcp=Cp
1220  I=17
1225  K=18
1230  D_log=.027
1235  FOR J=10 TO 200
1240  Freqen=10^(-1+D_log*J)
```

Main Program Listing—continued.

```
1245  FOR L=K TO Cp
1250  K=L
1255  IF Frc(K)<Freqen THEN Nextl
1260  I=I+1
1265  Ryc(I)=Ryc(K)
1270  Ixc(I)=Ixc(K)
1275  Frc(I)=Frc(K)
1280  GOTO Nextj
1285 Nextl:  NEXT L
1290 Nextj:  NEXT J
1295  Cp=I
1300  FOR J=Cp+1 TO Oldcp
1305  Ryc(J)=Ixc(J)=Frc(J)=0
1310  NEXT J
1315 SUBEND
1320  !
1325 ! SUBROUTINE TO SET MODE
1330 ! CALL:     CALL MODE
1335 !
1340 !
1345 !
1350 SUB Mode
1355 OPTION BASE 1
1360 COM Command$(*),Humber_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
1365 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Hsave,Stat
1370 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
1375 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
1380 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
1385 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
1390 COM SHOPT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
1395 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
1400 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
1405 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
1410 CALL Clear
1415 Mode$=" "
1420 CALL Putline(0,26,"MODE CONTROL MENU")
1425 CALL Putline(4,10,"WHICH MODE DO YOU WISH TO OPERATE IN ?")
1430 ! CALL Putline(6,15,"I=IMMEDIATE EXECUTION OF COMMANDS")
1435 CALL Putline(7,15,"S=STORE COMMAHDS FOR EXECUTION LATER")
1440 CALL Putline(8,15,"X=EXECUTE STORED INSTRUCTIONS")
1445 CALL Putline(10,12,"INPUT YOUR CHOICE *")
1450 Menu=5
1455 LINPUT Choice$
1460 IF Choice$="I" THEN Mode$="IMM"
1465 IF Choice$="S" THEN Mode$="STO"
1470 IF Choice$="X" THEN Mode$="EXE"
1475 IF Choice$="0" THEN 1495
1480 IF Mode$=" " THEN 1410
1485 IF Mode$="EXE" THEN CALL Send
1490 WAIT 2000
1495 SUBEND
1500 !
1505 !
1510 SUB Key_6
1515 OPTION BASE 1
1520 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
1525 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Hsave,Stat
1530 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
1535 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
1540 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
1545 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
```

Main Program Listing--continued.

```
1550 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
1555 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
1560 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
1565 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
1570 LINK "KEY_6:F9",13420
1575  CALL Command_file
1580  D9=6
1585  SUBEND
1590 !
1595 !
1600 !
1605 ! SUBROUTINE TO PERFORM EDITING
1610 ! CALL:     CALL EDITL
1615 SUB Editl
1620 OPTION BASE 1
1625 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
1630 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
1635 COM Files,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
1640 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
1645 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
1650 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
1655 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
1660 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
1665 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
1670 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
1675 Begin: !
1680 Editf$="EDIT"
1685 CALL Clear
1690 CALL Putline(0,26,"EDITING FUNCTION MENU")
1695 CALL Putline(2,5,"YOU CAN DO THE FOLLOWING EDITING:")
1700 CALL Putline(4,10,"A=ADD    D=DELETE    R=REPLACE    E=EXIT   L=LIST   S=SYNTHES
IZER")
1705 CALL Putline(6,5,"INPUT YOUR CHOICE *")
1710 INPUT Choice$
1715 IF Choice$="A" THEN Add_lines
1720 IF Choice$="D" THEN Delete_lines
1725 IF Choice$="R" THEN Add_lines
1730 IF Choice$="E" THEN 2025
1735  IF (Choice$="Q") OR (Choice$="S") THEN Add_lines
1740  IF Choice$="L" THEN CALL C_dmp
1745 IF Choice$="0" THEN 2025
1750 BEEP
1755 GOTO 1685
1760 Add_lines: !
1765 IF (Choice$="A") OR (Choice$="Q") OR (Choice$="S") THEN Editf$="ADD"
1770 IF Choice$="R" THEN Editf$="REP"
1775 CALL Putline(9,5,"AT WHAT COMMAND DO YOU WISH TO ADD/REPLACE *")
1780 INPUT Edit_line
1785  IF Edit_line<=0 THEN GOTO Begin
1790  IF (Choice$="Q") OR (Choice$="S") THEN 1800
1795  GOTO 1875
1800  CALL Shifty(1)
1805  PRINT "COMMAND$=?"
1810  LINPUT Command$(Edit_line-1)
1815  IF Choice$="S" THEN Prefix$(Edit_line-1)="3325A"
1820  IF Choice$="S" THEN 1835
1825  PRINT "PREFIX$=?"
1830  LINPUT Prefix$(Edit_line-1)
1835  IF Prefix$(Edit_line-1)="3325A" THEN Word$(Edit_line-1)="** SYNTHESIZER **
"
1840  IF Prefix$(Edit_line-1)="3325A" THEN 1855
1845  PRINT "WORD$=?"
1850  LINPUT Word$(Edit_line-1)
```

Main Program Listing—continued.

```
1855  IF Edit_line>Number_commands THEN Number_commands=Number_commands+1
1860  IF Editf$="ADD" THEN Number_commands=Number_commands+1
1865  GOTO Begin
1870 CALL Putline(11,7,"YOU MAY ADD/REPLACE THE FOLLOWING TYPES:")
1875 CALL Putline(13,10,"1=SETUP        2=CURSOR        3=DISPLAY        4=CONTROL"
)
1880 CALL Putline(14,10,"5=SPECIAL      6=COMPUTER      7=DATA FILES     8=DATA MAN
IP. ")
1885  CALL Putline(15,10,"9=PLOT LIMITS")
1890 CALL Putline(16,14,"INPUT YOUR CHOICE*")
1895 INPUT Group
1900  IF Group<=0 THEN GOTO Begin
1905  IF Group<>5 THEN D9=-1
1910 IF Group=1 THEN CALL Key_0
1915 IF Group=2 THEN CALL Key_1
1920 IF Group=3 THEN CALL Key_2
1925 IF Group=4 THEN CALL Key_3
1930 IF Group=5 THEN CALL Key_4
1935  IF Group=7 THEN CALL Key_7
1940  IF (Group=6) AND (Choice$<>"R") THEN CALL Key_12
1945  IF (Group=6) AND (Choice$="R") THEN 1980
1950  IF (Group=8) AND (Choice$<>"R") THEN CALL Key_8
1955  IF (Group=8) AND (Choice$="R") THEN 1980
1960  IF (Group=9) AND (Choice$<>"R") THEN CALL Key_14
1965  IF (Group=9) AND (Choice$="R") THEN 1980
1970 ! IF Group<>6 THEN 6700
1975  GOTO 2000
1980  BEEP
1985  PRINT "REPLACE NOT AVAILABLE WITH GROUPS #6,8,9 -- SORRY"
1990  WAIT 2000
1995  GOTO 1685
2000 IF (Group>=1) AND (Group<=9) THEN 2015
2005 BEEP
2010 GOTO 1870
2015 IF Editf$="REP" THEN Number_commands=Number_commands-1
2020 GOTO Begin
2025 Editf$=" "
2030 GOTO Bye
2035 ! SUBEND
2040 Delete_lines: !
2045 CALL Putline(9,5,"ENTER LINES TO BE DELETED*")
2050 LINPUT D1$
2055  P=POS(D1$,",")
2060  IF P=0 THEN Start1=VAL(D1$)
2065  IF P<>0 THEN Start1=VAL(D1$[1,P-1])
2070  IF P<>0 THEN Stop1=VAL(D1$[P+1])
2075  IF P=0 THEN Stop1=Start1
2080  IF (Start1<=0) OR (Stop1<=0) THEN GOTO Begin
2085  IF Stop1>=Start1 THEN GOTO 2105
2090 BEEP
2095 PRINT "TURKEY -- ENTER LINE NUMBERS CORRECTLY"
2100 GOTO Begin
2105  Number=Stop1-Start1+1
2110 FOR N=Start1 TO Number_commands
2115  IF N+Number>200 THEN 2140
2120 Command$(N)=Command$(N+Number)
2125 Prefix$(N)=Prefix$(N+Number)
2130 Word$(N)=Word$(N+Number)
2135 NEXT N
2140 Number_commands=Number_commands-Number
2145 GOTO Begin
2150 Bye: !
2155 Menu=9
2160 SUBEND
2165 !
2170 !
2175 !
```

Main Program Listing—continued.

```
2180 !
2185 ! 5420 INTERRUPT HANDLING SUBROUTINE
2190 !
2195 SUB Isr
2200 OPTION BASE 1
2205 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
2210 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
2215 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
2220 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
2225 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
2230 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
2235 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
2240 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
2245 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
2250 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
2255 STATUS A5420;Stat
2260 ! STATUS 705;Pstat
2265 Stat=OCTAL(Stat)
2270 IF Ti=0 THEN 2290
2275 PRINTER IS 0
2280 PRINT Stat,Pstat
2285 PRINTER IS 16
2290 IF Stat=146 THEN CALL Print_data
2295 IF Stat=150 THEN CALL Make_plot
2300 IF Stat=142 THEN CALL Make_plot
2305 IF Stat=170 THEN CALL Make_plot
2310 IF (Stat=140) OR (Stat=160) THEN CALL Savrec
2315 IF (Stat=141) OR (Stat=161) THEN B5420=1
2320 IF (Stat=104) OR (Stat=105) THEN B5420=0
2325 IF (Stat=106) OR (Stat=107) OR (Stat=110) THEN CALL C_err
2330 IF Stat=101 THEN CALL Adc_ovrflo
2335 CARD ENABLE 7
2340 SUBEND
2345 !
2350 !
2355 !
2360 !
2365 SUB Dummyf
2370 SUBEND
2375 !
2380 !
2385 !
2390 !
2395 ! SUBROUTINE TO PLOT DATA
2400 ! CALL     CALL MAKE_PLOT
2405 !
2410 !
2415 SUB Make_plot
2420  OPTION BASE 1
2425 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
2430 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
2435 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
2440 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
2445 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
2450 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
2455 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
2460 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
2465 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
2470 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
2475 IF Stat=150 THEN Mplot
```

Main Program Listing—continued.

```
2480 IF Stat=170 THEN Nplot
2485 B5420=0
2490 GOTO Bye
2495 Mplot: !
2500 B5420=1
2505 CONFIGURE 7 TALK = 4 LISTEN = 5
2510 SENDBUS 7;"?D%"
2515 GOTO Bye
2520 Nplot: !
2525 B5420=1
2530 CONFIGURE 7 TALK = 5 LISTEN = 4
2535 SENDBUS 7;"?E$"
2540 Bye: !
2545 SUBEND
2550 !
2555 !
2560 !
2565 !
2570 ! SUBROUTINE TO DELETE COMMAND STACK
2575 ! CALL:     CALL DEL_COM
2580 !
2585 !
2590 SUB Del_com
2595  OPTION BASE 1
2600 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
2605 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
2610 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
2615 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
2620 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
2625 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
2630 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device.Pager.Nopager
2635 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
2640 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
2645 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
2650 CALL Clear
2655 CALL Putline(3,26,"COMMANDS BEING DELETED")
2660 FOR J=1 TO Number_commands
2665 Command$(J)=""
2670 Prefix$(J)=""
2675 Word$(J)=""
2680 NEXT J
2685 Number_commands=0
2690 WAIT 500
2695 Menu=10
2700 SUBEND
2705 !
2710 !
2715 !
2720 !
2725 !
2730 ! SUBROUTINE FOR SAVE - RECALL OF DATA
2735 ! CALL:     CALL SAVREC
2740 !
2745 !
2750 !
2755 SUB Savrec
2760 OPTION BASE 1
2765 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
2770 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
2775 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
2780 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
2785 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
```

Main Program Listing—continued.

```
2790 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
2795 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
2800 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
2805 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
2810 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
2815  SHORT Tempa(16)
2820  ON ERROR GOTO Trouble
2825  GOTO 2845
2830 Trouble:  IF ERRN=153 THEN 2920
2835  PRINT "SAV/REC DATA ERROR: ";ERRN
2840  STOP
2845 IF Stat=140 THEN Readin
2850 IF Stat=160 THEN Sendout
2855 Readin: !
2860  !
2865  MAT Sdata=ZER
2870  MAT Ry=ZER
2875  MAT Ix=ZER
2880 FOR J=1 TO 16
2885 ENTER A5420;Tempa(J)
2895 NEXT J
2900 Dt=BINAND(Tempa(5),3)
2905 IF Dt=2 THEN Np=Tempa(3)/2
2910 IF Dt=3 THEN Np=Tempa(3)/4
2915  REDIM Sdata(Np*2)
2920  OVERLAP
2925 ENTER 704 USING "+,F";Sdata(*)
2926  REDIM Sdata(528)
2930  IF Dt=2 THEN Dt=1
2935  IF Dt=3 THEN Dt=0
2945  IF Dt=0 THEN Cmplx
2950  FOR J=1 TO Np
2955  Ry(J)=Sdata(J)
2960  NEXT J
2965  GOTO Xx
2970 Cmplx:  !
2975  Index=0
2980  FOR J=1 TO Np
2985  Index=Index+1
2990  Ry(J)=Sdata(Index)
2995  Index=Index+1
3000  Ix(J)=Sdata(Index)
3005  NEXT J
3010 Xx:  !
3015  Xstart=Tempa(12)
3020  Xstep=Tempa(13)
3021  FOR J=513 TO 528
3022  Sdata(J)=Tempa(J-512)
3023  NEXT J
3025  SERIAL
3030 GOTO Bye
3035 !
3040 Sendout: !
3045 Upper=16+Sdata(3)/2
3050 FOR J=1 TO Upper
3055 OUTPUT 704;Sdata(J)
3060 NEXT J
3065 !
3070 Bye: !
3075 B5420=0
3080 SUBEND
3085 !
3090 !
3095 SUB Adc_ovrflo
```

Main Program Listing—continued.

```
3100 FOR J=1 TO 1
3105 BEEP
3110 WAIT 500
3115 NEXT J
3120 ! STOP
3125 SUBEND
3130 !
3135 !
3140 !
3145 !
3150 ! SUBROUTINE TO INDICATE COMMAND ERROR
3155 ! CALL:     CALL C_ERR
3160 SUB C_err
3165 CALL Clear
3170 IF Stat=106 THEN CALL Putline(2,25,"FATAL COMMAND ERROR")
3175 IF Stat=107 THEN CALL Putline(3,25,"FATAL GENERAL ERROR")
3180 IF Stat=110 THEN CALL Putline(4,25," FATAL ERROR! ")
3185 SUBEND
3190 !
3195 !
3200 !
3205 SUB Key_11
3210  OPTION BASE 1
3215  LINK "KEY_11:F9",13420
3220  CALL Mul_file
3225  SUBEND
3230 !
3235  !
3240  !
3245  !
3250  SUB Key_12
3255  OPTION BASE 1
3260  LINK "KEY_12:F9",13420
3265  CALL Com_group
3270  SUBEND
3275  !
3280  !
3285  SUB Clear_line(Istart,Numl)
3290  FOR J=Istart TO Istart+Numl-1
3295  PRINT USING "#,3A,2D,K";CHR$(27)&"&a",J,"r0C"&CHR$(27)&"K"
3300  NEXT J
3305  SUBEND
3310  !
3315  !
3320  SUB Get_p(Parm$,L)
3325  OPTION BASE 1
3330 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
3335 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
3340 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
3345 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
3350 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
3355 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
3360 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
3365 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
3370 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
3375 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
3380  Number_commands=Number_commands+1
3385  Parm$=" "
3390  PRINT "ANY PARAMETERS (Y OR N) ? "
3395  L=Number_commands
3400  INPUT Answer$
3405  IF Answer$="N" THEN Bye
3410  PRINT "INPUT PARAMETER STRING IN QUOTES:"
```

Main Program Listing--continued.

```
3415  LINPUT Parm$
3420  Parm$=Parm$[2,LEN(Parm$)-1]
3425 Bye: !
3430  SUBEND
3435  !
3440  !
3445  SUB Shifty(N)
3450  OPTION BASE 1
3455 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
3460 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
3465 COM Files,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
3470 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
3475 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
3480 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
3485 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
3490 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
3495 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut.Time$[
20]
3500 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
3505  P=1
3510  ! PRINT Number_commands,Edit_line
3515  FOR H=1 TO N
3520  FOR K=Number_commands+P TO Edit_line+P STEP -1
3525  Command$(K)=Command$(K-1)
3530  Word$(K)=Word$(K-1)
3535  Prefix$(K)=Prefix$(K-1)
3540  NEXT K
3545  P=P+1
3550  NEXT H
3555  Edit_line=Edit_line+N
3560  SUBEND
3565  !
3570  !
3575  !
3580  !
3585  !
3590  SUB Group_menu(R_start,R_stop,Pos_r,Pos_s,Delt,Nlbl,English$(*))
3595  Count=0
3600  FOR R=R_start TO R_stop
3605  FOR K=1 TO Pos_r
3610  Count=Count+1
3615  IF Count>Nlbl THEN Bye
3620  J=Pos_s+(K-1)*Delt
3625  IF Count<10 THEN Line$=VAL$(Count)&"  =  "&English$(Count)
3630  IF Count>9 THEN Line$=VAL$(Count)&" =  "&English$(Count)
3635  CALL Putline(R,J,Line$)
3640  NEXT K
3645  NEXT R
3650 Bye:   !
3655  SUBEND
3660  !
3665  !
3670  SUB Label(Xsize,Ysize,Xlabel,Ylabel,Size,Pen,L$)
3675  INTEGER Xo,Yo,Xmax,Ymax,Labelx,Labely,Npen
3680  Npen=Pen
3685  Xo=0
3690  Yo=0
3695  Xmax=16000*Xsize/15.75
3700  Ymax=11400*Ysize/11.2
3705  Labelx=Xlabel*1016-.5*1016
3710  Labely=Ylabel*1016-.25*1016
3715  Chrsiz=Size/64*2.5
3720  IMAGE 3A,5DC,5DC,5DC,5D
3725  OUTPUT 705 USING 3720;"IP ",Xo,Yo,Xmax,Ymax
```

Main Program Listing—continued.

```
3730  OUTPUT 705 USING 3720;"IW ",Xo,Yo,Xmax,Ymax
3735  OUTPUT 705;"PU"
3740  OUTPUT 705 USING 3720;"PA ",Xo,Yo,Labelx,Labely
3745  OUTPUT 705 USING "3A,K,A,K";"SI ",Chrsiz*(5/8),",",Chrsiz
3750  OUTPUT 705 USING "3A,D";"SP ",Npen
3755  OUTPUT 705;"LB"&L$&CHR$(3)
3760  OUTPUT 705;"SP 0"
3765  SUBEND
3770  !
3775  !
3780  SUB Sys(Code)
3785  OPTION BASE 1
3790 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
3795 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
3800 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
3805 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
3810 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
3815 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
3820 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
3825 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
3830 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
3835 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
3840  SHORT Xar(1536),Yar(1536)
3845  IF (Code=1) OR (Code=2) THEN R_label
3850  IF (Code)=3) AND (Code<=8) THEN Plots
3855  IF (Code=9) OR (Code=10) THEN CALL Data_file(Code)
3860  IF (Code=18) OR (Code=19) THEN CALL Data_file(Code)
3865  IF Code=20 THEN CALL Data_file(Code)
3870  IF Code=21 THEN CALL Data_file(Code)
3875  IF Code=11 THEN CALL Tie(Word$(Xline))
3880  IF (Code)=12) AND (Code<=17) THEN Plots
3885  IF (Code=24) OR (Code=40) THEN Plots
3890  IF (Code)=22) AND (Code<=23) THEN CALL Datam(Code,Word$(Xline))
3895  IF (Code)=26) AND (Code<=29) THEN CALL Plot_fix(Code)
3900  IF (Code)=32) AND (Code<=37) THEN CALL Smoothr(Code)
3905  IF (Code=30) OR (Code=31) THEN CALL Splot(Code,Word$(Xline))
3910  IF (Code=41) OR (Code=42) THEN CALL Splot(Code,Word$(Xline))
3915  IF Code=38 THEN CALL Movr
3920  IF Code=39 THEN CALL Del_files(Word$(Xline))
3925  IF Code=43 THEN CALL Analysis(Code,N,Xar(*),Yar(*))
3930  IF Code=44 THEN CALL Remove(Code,D1,Frequency,Every,X60,Cp,Ryc(*),Ixc(*),F
rc(*))
3935  IF Code=45 THEN D5=1   ! ENABLE ANALYSIS ON EXPERIMENTAL DATA
3940  IF Code=46 THEN CALL Shorty(Cp,Ryc(*),Ixc(*),Frc(*))
3945  GOTO Bye
3950 R_label: !
3955  Xline=Xline+1
3960  P=POS(Command$(Xline),",")
3965  X_size=VAL(Command$(Xline)[1,P-1])
3970  Y_size=VAL(Command$(Xline)[P+1])
3975  P=POS(Prefix$(Xline),",")
3980  Label_x=VAL(Prefix$(Xline)[1,P-1])
3985  Label_y=VAL(Prefix$(Xline)[P+1])
3990  Xline=Xline+1
3995  P=POS(Prefix$(Xline),",")
4000  Size=VAL(Prefix$(Xline)[1,P-1])
4005  Pentype=VAL(Prefix$(Xline)[P+1])
4010  IF Code=1 THEN L$=Command$(Xline)
4015  IF Code=2 THEN L$=File$
4020  ! PRINT X_size;Y_size;Label_x;Label_y;Size;Pentype;L$
4025  CALL Label(X_size,Y_size,Label_x,Label_y,Size,Pentype,L$)
4030  GOTO Bye
4035 Plots:  !
4040  Xline=Xline+1
```

Main Program Listing—continued.

```
4045  X$=Command$(Xline)
4050  Y$=Prefix$(Xline)
4055  T$=Word$(Xline)
4060  Xline=Xline+1
4065  C=Code
4070  Q=POS(Prefix$(Xline),",")
4075  Pn=VAL(Prefix$(Xline)[1,Q-1])
4080  P=VAL(Prefix$(Xline)[Q+1])
4085  CALL Abuild(C,N,Xar(*),Yar(*))
4090  IF (C>=3) AND (C<=5) THEN CALL Semilog(C,N,Xar(*),Yar(*),X$,Y$,T$,P,Pn)
4095  IF (C>=6) AND (C<=8) THEN CALL Linearc(C,N,Xar(*),Yar(*),X$,Y$,T$,P,Pn)
4100  IF (C>=12) AND (C<=18) THEN CALL Linearc(C,N,Xar(*),Yar(*),X$,Y$,T$,P,Pn)
4105  IF (C=24) OR (C=40) THEN CALL Linearc(C,N,Xar(*),Yar(*),X$,Y$,T$,P,Pn)
4110  GOTO Bye
4115 Bye: !
4120  SUBEND
4125  !
4130  !
4135  !
4140  !
4145  SUB Dummyh
4150  SUBEND
4155  !
4160  !
4165  !
4170  SUB Movr
4175  OPTION BASE 1
4180 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
4185 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
4190 COM Files,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
4195 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
4200 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
4205 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
4210 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
4215 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
4220 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
4225 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
4230  Cp=Np-1
4235  FOR J=2 TO Cp
4240  Ryc(J)=Ry(J)
4245  Ixc(J)=Ix(J)
4250  Frc(J)=Xstart+Xstep*(J-1)
4255  NEXT J
4260  SUBEND
4265  !
4270  !
4275  !
4280  !
4285  !
4290  !
4295  !
4300  ! SUBROUTINE TO STORE AND RECALL DATA
4305  ! CALL:     CALL DATA_FILE
4310  !
4315  SUB Data_file(Code)
4320  OPTION BASE 1
4325 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
4330 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
4335 COM Files,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
4340 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
4345 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
4350 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
```

Main Program Listing—continued.

```
8,D9
4355 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
4360 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
4365 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
4370 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
4375  ON ERROR GOTO Trouble
4380  IF (Code=9) OR (Code=18) THEN Save_data
4385  IF (Code=10) OR (Code=19) THEN Recall_data
4390  IF Code=20 THEN Zero_data
4395  IF Code=21 THEN Ren_fl
4400  PRINT "BAD CODE IN DATA_FILE: ";Code
4405  STOP
4410 Trouble:  !
4415  IF ERRN=54 THEN Dup_file
4420  IF ERRN=56 THEN No_file
4425  PRINT "HUH - DATA_FILE ERROR # : ",ERRN
4430  STOP
4435 No_file:  !
4440  BEEP
4445  PRINT "FILE ";Name$;" DOES NOT EXIST -- HALT!"
4450  STOP
4455 Dup_file:  !
4460  IF Code=18 THEN PURGE Name$
4465  GOTO 4645
4470 Zero_data:  !
4475  CALL Zero_data_sets
4480  GOTO Bye
4485 Recall_data:  ! 10=SHORT  19=TIED
4490  Name$=Word$(Xline)
4495  IF POS(Name$,":")=0 THEN Name$=Name$&":F8"
4500  ASSIGN #5 TO Name$
4505  IF Code=10 THEN READ #5;Np,Xstart,Xstep
4510  IF Code=19 THEN 4565
4515  IF Np<>0 THEN 4555
4520  Xstart=0
4525  Xstep=0
4530  MAT Ry=ZER
4535  MAT Ix=ZER
4540  MAT Sdata=ZER
4545  MAT Pdata=ZER
4550  GOTO 4600
4555  IF Code=10 THEN MAT READ #5;Ry,Ix,Sdata  ! ,Pdata
4560  IF Code=10 THEN 4600
4565  IF Code=19 THEN READ #5;Cp
4570  IF Cp<>0 THEN 4595
4575  MAT Ryc=ZER
4580  MAT Ixc=ZER
4585  MAT Frc=ZER
4590  GOTO 4600
4595  IF Code=19 THEN MAT READ #5;Ryc,Ixc,Frc
4600  ASSIGN * TO #5
4605  D1=0
4610  Code1=0
4615  CALL Remove(Code1,D1,Frequency,Every,X60,Cp,Ryc(*),Ixc(*),Frc(*))
4620  GOTO Bye
4625 Save_data:  ! 9=SHORT  18=TIED
4630  Name$=Word$(Xline)
4635  IF POS(Name$,":")=0 THEN Name$=Name$&":F8"
4640  IF Code=9 THEN CREATE Name$,26
4645  IF Code=18 THEN CREATE Name$,75
4650  ASSIGN #5 TO Name$
4655  IF Code=9 THEN PRINT #5;Np,Xstart,Xstep
4660  IF Np=0 THEN 4670
4665  IF Code=9 THEN MAT PRINT #5;Ry,Ix,Sdata  ! ,Pdata
4670  IF Code=18 THEN PRINT #5;Cp
```

Main Program Listing—continued.

```
4675  IF Cp=0 THEN 4685
4680  IF Code=18 THEN MAT PRINT #5;Ryc,Ixc,Frc
4685  ASSIGN * TO #5
4690  GOTO Bye
4695 Ren_f1:  !
4700  Name$=Word$(Xline)
4705  CALL Ren_file(Name$)
4710  GOTO Bye
4715 Bye:  !
4720  SUBEND
4725  !
4730  !
4735  !
4740  SUB Key_7
4745  OPTION BASE 1
4750  LINK "KEY_7:F9",13420
4755  CALL Dfile
4760  SUBEND
4765  !
4770  !
4775  !
4780  SUB Tie(F$)
4785  OPTION BASE 1
4790 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
4795 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
4800 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
4805 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
4810 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
4815 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
4820 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
4825 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
4830 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
4835 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
4840  DIM Bp(7)
4845  DATA 0.05, 0.1, 3, 25, 256, 3200, 25600
4850  MAT READ Bp ! FREQUENCY BREAKPOINTS
4855  DIM Data_files(6)
4860  DATA "1", "2", "3", "4", "5", "6"
4865  MAT READ Data_files  ! DATA FILES
4870  Cp=0
4875  Suf$="X"
4880  IF POS(F$,"X")<>0 THEN Suf$="X"
4885  IF POS(F$,"S")<>0 THEN Suf$="S"
4890  IF POS(F$,"C")<>0 THEN Suf$="C"
4895  FOR J=1 TO 6
4900  ! IF POS(F$,VAL$(J))=0 THEN Next_file
4905  Data_files$(J)=Data_files$(J)&Suf$&":F8"
4910  ASSIGN #5 TO Data_files$(J)
4915  READ #5;Np,Xstart,Xstep
4920  IF Np=0 THEN 4930
4925  MAT READ #5;Ry,Ix ! ,Sdata,Pdata
4930  ASSIGN * TO #5
4935  S=1
4940  IF Np>256 THEN S=2
4945  FOR K=1 TO 256 STEP S
4950  Freq=Xstart+Xstep*(K-1)
4955  IF (Freq<Bp(J)) OR (Freq>Bp(J+1)) THEN Skip
4960  IF Cp=0 THEN 4970
4965  IF (K>1) AND (Freq=Frc(Cp)) THEN Skip
4970  Cp=Cp+1
4975  Ryc(Cp)=Ry(K)
4980  Ixc(Cp)=Ix(K)
4985  Frc(Cp)=Freq
```

Main Program Listing—continued.

```
4990 Skip:  !
4995  NEXT K
5000 Next_file:  !
5005  NEXT J
5010 Bye:  !
5015  SUBEND
5020  !
5025  SUB Linearc(Code,N,SHORT X_array(*),Y_array(*),X$,Y$,T$,P1,Pn)
5030  OPTION BASE 1
5035 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
5040 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
5045 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Hp,Slope,P1,P2
5050 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
5055 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
5060 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
5065 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
5070 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
5075 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
5080 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
5085  Sflg=0
5090  D=1.! STARTING POINT IN DATA
5095  ! TRACE VARIABLES Pflg,Sflg,N,Upper,D
5100  Upper=N
5105   IF (Pflg=1) AND (Region=1) THEN D=Pstart
5110   IF Lsf=2 THEN D=1
5115  X$=TRIM$(X$)
5120  Y$=TRIM$(Y$)
5125  T$=TRIM$(T$)
5130  ! ARBITRARILY SET YMIN & YMAX
5135  IF Paxis=0 THEN GCLEAR
5140  IF Paxis>=2 THEN Pscale=0
5145  IF Pscale=0 THEN 5255  ! FIXED PLOT LIMITS
5150  Axmin=BIT(Pscale,0)
5155  Axmax=BIT(Pscale,1)
5160  Aymin=BIT(Pscale,2)
5165  Aymax=BIT(Pscale,3)
5170  IF Aymin THEN Ymin=Y_array(D)
5175  IF Aymax THEN Ymax=Y_array(D)
5180  IF Axmin THEN Xmin=X_array(D)
5185  IF Axmax THEN Xmax=X_array(D)
5190  ! NOW FIND MAX & MIN VALUES
5195  Upper=N
5200  IF (Pflg=1) AND (Region=1) THEN Upper=Pstop
5205  IF Lsf=2 THEN Upper=128
5210  FOR J=D TO Upper
5215  P=X_array(J)
5220  Q=Y_array(J)
5225  IF Axmin THEN Xmin=MIN(P,Xmin)
5230  IF Axmax THEN Xmax=MAX(P,Xmax)
5235  IF Aymin THEN Ymin=MIN(Q,Ymin)
5240  IF Aymax THEN Ymax=MAX(Q,Ymax)
5245  NEXT J
5250  !
5255  IF (Paxis=2) OR (Paxis=3) THEN 5285
5260  IF P1=1 THEN PLOTTER IS "GRAPHICS"
5265  IF P1=2 THEN PLOTTER IS "GRAPHICS"
5270  IF P1=3 THEN PLOTTER IS "9872A"
5275  !
5280  ! SET PLOT LIMITS
5285  T=1.5
5290  LINE TYPE 1
5295  CALL Plfix
5300  PEN Pn
5305  FRAME
```

Main Program Listing—continued.

```
5310  GRAPHICS
5315  ! PLOT DATA
5320  FOR J=D TO Upper
5325  X=X_array(J)
5330  Y=Y_array(J)
5335  IF J=D THEN PLOT X,Y,-2
5340  IF Code<>16 THEN LINE TYPE 2
5345  PLOT X,Y,-1
5350  NEXT J
5355  LINE TYPE 1
5360  IF (Paxis=2) OR (Paxis=3) THEN 5580  ! 5350
5365  SETGU
5370  !
5375  ! PUT LABELS ON Y-AXIS
5380  Ystep=(Ymax-Ymin)/4
5385  Ydel=(Ym-Yo)/4
5390  CSIZE 3.1
5395  LORG 2
5400  FOR J=1 TO 5
5405  IF J=1 THEN MOVE Xo,Yo
5410  Yfac=Ydel*(J-1)+Yo
5415  PLOT Xo,Yfac,-2
5420  PLOT Xo-T,Yfac,-1
5425  PLOT Xo-20,Yfac,-2
5430  LABEL USING "SD.DDE";Ymin+Ystep*(J-1)
5435  NEXT J
5440  ! PUT LABELS ON X AXIS
5445  LORG 5
5450  Xstp=(Xmax-Xmin)/3
5455  Xdel=(Xm-Xo)/3
5460  FOR J=1 TO 4
5465  IF J=1 THEN MOVE Xo,Yo
5470  Xfac=Xdel*(J-1)+Xo
5475  PLOT Xfac,Yo,-2
5480  PLOT Xfac,Yo-T,-1
5485  PLOT Xfac,Yo-6,-2
5490  LABEL USING "SD.DDE";Xmin+Xstp*(J-1)
5495  NEXT J
5500  !
5505  ! PUT LABELS ON PLOT
5510  Xmid=Xo+(Xm-Xo)/2
5515  Ymid=Yo+(Ym-Yo)/2
5520  ! PRINT #0;Xmid
5525  CSIZE 4.2
5530  LDIR 0
5535  LORG 5
5540  PLOT Xmid,Yo-12,-2
5545  LABEL USING "K";X$
5550  PLOT Xmid,Ym+5,-2
5555  LABEL USING "K";T$
5560  DEG
5565  LDIR 90
5570  PLOT Xo-25,Ymid,-2
5575  LABEL USING "K";Y$
5580  IF (Region=1) AND (Pflg=0) THEN CALL Region_intrest(N,X_array(*),Y_array(*
))
5585  IF D5=1 THEN CALL Analysis(Code,N,X_array(*),Y_array(*))
5590  IF Pflg=2 THEN 5715
5595  ! IF Sflg=1 THEN CALL Unswap_scale(N,X_array(*),Y_array(*))
5600  IF Cur=1 THEN SCALE Xmin,Xmax,Ymin,Ymax
5605  CALL Points(N,Cur,Pflg,Pstart,Pstop,X_array(*),Y_array(*),Frc(*))
5610  Maxdeg=15
5615  Desdeg=15
5620  IF Poly=1 THEN CALL Fit(Maxdeg,Desdeg,N,X_array(*),Y_array(*))
5625  IF Lsf<>0 THEN CALL Lsqfit(Code,N,X_array(*),Y_array(*))
5630  IF (Sflg=2) AND (Region=1) THEN 5650
```

Main Program Listing—continued.

```
5635  IF Sflg=2 THEN 5650
5640  IF Sflg<0 THEN 5650
5645  IF Sflg=1 THEN GOTO 5090
5650  IF Paper=Blue THEN 5670
5655  PRINTER IS 0
5660  IF (Paxis=0) OR (Paxis=3) THEN PRINT PAGE
5665  PRINTER IS 16
5670  IF (Paxis=0) AND (P1=2) THEN DUMP GRAPHICS
5675  IF (Paxis=3) AND (P1=2) THEN DUMP GRAPHICS
5680  IF (Paxis=0) OR (Paxis=3) THEN GCLEAR
5685  IF (Paxis=0) OR (Paxis=3) THEN EXIT GRAPHICS
5690  PEN 0
5695  Cur=0
5700  Poly=0
5705  Lsf=0
5710  D5=0
5715  IF Pflg<>2 THEN 5725
5720  Pflg=1
5725  SUBEND
5730  !
5735  !
5740  !
5745  !
5750  SUB Key_8
5755  OPTION BASE 1
5760  LINK "KEY_8:F9",13420
5765  CALL Dmanip
5770  SUBEND
5775  !
5780  !
5785  !
5790  !
5795  SUB Datam(Code,S$)
5800  OPTION BASE 1
5805 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
5810 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Hprint,Sdata(*),Nsave,Stat
5815 COM Files$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
5820 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
5825 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
5830 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
5835 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
5840 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
5845 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
5850 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
5855  IF Code=22 THEN Swap
5860  IF Code=23 THEN Scales
5865  PRINT "BAD CODE ";Code
5870  STOP
5875 Swap:  !
5880  FOR J=1 TO Np
5885  T=Ry(J)
5890  Ry(J)=Ix(J)
5895  Ix(J)=T
5900  NEXT J
5905  FOR J=1 TO Cp
5910  T=Ryc(J)
5915  Ryc(J)=Ixc(J)
5920  Ixc(J)=T
5925  NEXT J
5930  GOTO Bye
5935 Scales:  !
5936  Rflg=Iflg=0
5937  IF POS(S$,"R") THEN Rflg=1
```

Main Program Listing—continued.

```
5938  IF POS(S$,"I") THEN Iflg=1
5939  IF (Rflg=0) AND (Iflg=0) THEN Rflg=Iflg=1
5940  P=POS(S$,",")
5945  A$=S$[1,P-1]
5950  Fac=VAL(S$[P+1])
5955  IF A$="*" THEN Mults
5960  IF A$="/" THEN Divs
5965  PRINT "BAD A$ ";A$
5970  STOP
5975 Mults:  !
5980  IF Rflg THEN MAT Ry=Ry*(Fac)
5985  IF Iflg THEN MAT Ix=Ix*(Fac)
5990  IF Rflg THEN MAT Ryc=Ryc*(Fac)
5995  IF Iflg THEN MAT Ixc=Ixc*(Fac)
6000  GOTO Bye
6005 Divs:  !
6010  IF Rflg THEN MAT Ry=Ry/(Fac)
6015  IF Iflg THEN MAT Ix=Ix/(Fac)
6020  IF Rflg THEN MAT Ryc=Ryc/(Fac)
6025  IF Iflg THEN MAT Ixc=Ixc/(Fac)
6030  GOTO Bye
6035 Bye:  !
6040  SUBEND
6045  !
6050  !
6055  !
6060  !
6065  SUB Abuild(C,N,SHORT X_array(*),Y_array(*))
6070  OPTION BASE 1
6075 COM Commands$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
6080 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
6085 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
6090 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
6095 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Floc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
6100 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
6105 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
6110 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
6115 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
6120 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
6125 C_3:  IF C<>3 THEN C_4  ! SEMILOG ARRAY PLOTS
6130 C3e:  FOR J=1 TO Np
6135  X_array(J)=Ix(J)
6140  Y_array(J)=Ry(J)
6145  NEXT J
6150  N=Np
6155  GOTO Bye
6160  !
6165 C_4:  IF C<>4 THEN C_5  ! SEMILOG Y vs XSTART*XSTEP
6170 C4e:  FOR J=1 TO Np
6175  X_array(J)=Xstart+Xstep*(J-1)
6180  Y_array(J)=Ry(J)
6185  NEXT J
6190  N=Np
6195  GOTO Bye
6200  !
6205 C_5:  IF C<>5 THEN C_6  ! SEMILOG Y vs X*(XSTART*XSTEP)
6210 C_5e:  FOR J=1 TO Np
6215  X_array(J)=Ix(J)*(Xstart+Xstep*(J-1))
6220  Y_array(J)=Ry(J)
6225  NEXT J
6230  N=Np
6235  GOTO Bye
6240  !
6245 C_6:  IF C<>6 THEN C_7  ! LINEAR ARRAY PLOT
```

Main Program Listing—continued.

```
6250  GOTO C3e
6255  !
6260 C_7:  IF C<>7 THEN C_8  ! LINEAR Y vs XSTART*XSTEP
6265  GOTO C4e
6270  !
6275 C_8:  IF C<>8 THEN C_12 ! LINEAR Y vs X*(XSTART*XSTEP)
6280  GOTO C5e
6285  !
6290 C_12:  IF C<>12 THEN C_13 ! LINEAR IM vs REAL
6295 C_12e:  FOR J=1 TO Cp
6300  X_array(J)=Ryc(J)
6305  Y_array(J)=-Ixc(J)
6310  NEXT J
6315  N=Cp
6320  GOTO Bye
6325  !
6330 C_13:  IF C<>13 THEN C_14 ! LINEAR REAL vs IM*HZ
6335  FOR J=1 TO Cp
6340  Y_array(J)=Ryc(J)
6345  X_array(J)=-Ixc(J)*Frc(J)
6350  NEXT J
6355  N=Cp
6360  GOTO Bye
6365  !
6370 C_14:  IF C<>14 THEN C_15 ! LINEAR REAL vs IM/HZ
6375  FOR J=1 TO Cp
6380  Y_array(J)=Ryc(J)
6385  X_array(J)=-Ixc(J)/Frc(J)
6390  NEXT J
6395  N=Cp
6400  GOTO Bye
6405  !
6410 C_15:  IF C<>15 THEN C_16 ! COHERENCE PLOT
6415 C15e:  FOR J=1 TO Cp
6420  Y_array(J)=Ryc(J)
6425  X_array(J)=Frc(J)
6430  NEXT J
6435  N=Cp
6440  GOTO Bye
6445  !
6450 C_16:  IF C<>16 THEN C_17 ! S/N
6455 C16e:  FOR J=1 TO Cp
6460  T=Ryc(J)
6465  IF (T=0) AND (J>1) THEN T=Ryc(J-1)
6470  IF (T=0) AND (J=1) THEN T=Ryc(J+1)
6475  IF T=0 THEN T=1E-10
6480  Y_array(J)=20*LGT(ABS(T))
6485  X_array(J)=LGT(Frc(J))
6490  NEXT J
6495  N=Cp
6500  GOTO Bye
6505  !
6510 C_17:  IF C<>17 THEN C_24 ! PHASE
6515  DEG
6520  FOR J=1 TO Cp
6525  ! Y_ARRAY(J)=ATN(IXC(J)/RYC(J))
6530  Angl=ATN(Ixc(J)/Ryc(J))
6535  IF (Ixc(J)>=0) AND (Ryc>=0) THEN Y_array(J)=Angl
6540  IF (Ixc(J)>=0) AND (Ryc(J)<=0) THEN Y_array(J)=90+(90+Angl)
6545  IF (Ixc(J)<0) AND (Ryc(J)<=0) THEN Y_array(J)=-180+Angl
6550  IF (Ixc(J)<=0) AND (Ryc(J)>=0) THEN Y_array(J)=Angl
6555  X_array(J)=LGT(Frc(J))
6560  NEXT J
6565  N=Cp
6570  GOTO Bye
6575  !
```

Main Program Listing--continued.

```
6580 C_24:  IF C<>24 THEN C40 ! LOG MAG vs LOG HZ
6585  FOR J=1 TO Cp
6590  X_array(J)=LGT(Frc(J))
6595  Y_array(J)=LGT(SQR(Ryc(J)^2+Ixc(J)^2))
6600  NEXT J
6605  N=Cp
6610  GOTO Bye
6615  !
6620 C40:  IF C<>40 THEN Bye
6625  Stp=2*.1/(Np-1)
6630  FOR J=1 TO Np
6635  Y_array(J)=Ry(J)
6640  X_array(J)=-.1+Stp*(J-1)
6645  NEXT J
6650  N=Np
6655  GOTO Bye
6660 Bye:  !
6665  SUBEND
6670  !
6675  !
6680  !
6685  !
6690  !
6695  SUB Plfix
6700  OPTION BASE 1
6705 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
6710 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
6715 COM Files,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
6720 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
6725 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
6730 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
6735 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
6740 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
6745 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
6750 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
6755  IF Plim<>0 THEN Pf2
6760  Pxo=0
6765  Pxm=184
6770  Pyo=0
6775  Pym=149
6780 Pf2:  IF Ploc<>0 THEN Pf3
6785  Xo=40
6790  Xm=109
6795  Yo=30
6800  Ym=80
6805 Pf3:  LIMIT Pxo,Pxm,Pyo,Pym
6810  LOCATE Xo,Xm,Yo,Ym
6815  CLIP Xo,Xm,Yo,Ym
6820  SCALE Xmin,Xmax,Ymin,Ymax
6825  !
6830  SUBEND
6835  !
6840  !
6845  !
6850  !
6855  SUB Key_14
6860  OPTION BASE 1
6865  LINK "KEY_14:F9",13420
6870  CALL Plot_group
6875  SUBEND
6880  !
6885  !
6890  !
6895  !
```

Main Program Listing—continued.

```
6900  SUB Plot_fix(Code)
6905  OPTION BASE 1
6910 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
6915 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
6920 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,F1,F2
6925 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
6930 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
6935 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
6940 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
6945 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
6950 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
6955 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
6960  Xline=Xline+1
6965  Pair1$=Command$(Xline)
6970  Pair2$=Prefix$(Xline)
6975  IF Code=28 THEN C28
6980  P=POS(Pair1$,",")
6985  T1=VAL(Pair1$[1,P-1])
6990  T2=VAL(Pair1$[P+1])
6995  P=POS(Pair2$,",")
7000  T3=VAL(Pair2$[1,P-1])
7005  T4=VAL(Pair2$[P+1])
7010  IF (T1=T2) AND (T2=T3) AND (T3=T4) AND (T4=0) THEN Unset
7015  IF Code<>26 THEN C27
7020  Plim=1
7025  Pxo=T1
7030  Pxm=T2
7035  Pyo=T3
7040  Pym=T4
7045 C27:  IF Code<>27 THEN C28
7050  Ploc=1
7055  Xo=T1
7060  Xm=T2
7065  Yo=T3
7070  Ym=T4
7075 C28:  IF Code<>28 THEN C29
7080  Pscale=0
7085  P=POS(Pair1$,",")
7090  A$=Pair1$[1,P-1]
7095  IF A$<>"*" THEN Xmin=VAL(A$)
7100  IF A$="*" THEN Pscale=Pscale+1
7105  A$=Pair1$[P+1]
7110  IF A$<>"*" THEN Xmax=VAL(A$)
7115  IF A$="*" THEN Pscale=Pscale+2
7120  P=POS(Pair2$,",")
7125  A$=Pair2$[1,P-1]
7130  IF A$<>"*" THEN Ymin=VAL(A$)
7135  IF A$="*" THEN Pscale=Pscale+4
7140  A$=Pair2$[P+1]
7145  IF A$<>"*" THEN Ymax=VAL(A$)
7150  IF A$="*" THEN Pscale=Pscale+8
7155  IF (Xmin=Xmax) AND (Xmax=Ymin) AND (Ymin=Ymax) AND (Ymax=0) THEN Pscale=15
7160 C29:  IF Code<>29 THEN Bye
7165  Paxis=T1
7170  GOTO Bye
7175 Unset:  !
7180  IF Code=26 THEN Plim=0
7185  IF Code=27 THEN Ploc=0
7190  IF Code=28 THEN Pscale=0
7195  IF Code=29 THEN Paxis=0
7200 Bye: !
7205  SUBEND
7210  !
```

Main Program Listing—continued.

```
7215  !
7220  !
7225  !
7230  SUB Smoothr(Code)
7235  OPTION BASE 1
7240 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
7245 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
7250 COM Files$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
7255 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
7260 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
7265 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
7270 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
7275 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
7280 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Times$[
20]
7285 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
7290  N=VAL(Word$(Xline))
7295  IF (Code=32) OR (Code=34) THEN CALL Smooth(N,Np,Ry(*))
7300  IF (Code=33) OR (Code=34) THEN CALL Smooth(N,Np,Ix(*))
7305  IF (Code=35) OR (Code=37) THEN CALL Smooth(N,Cp,Ryc(*))
7310  IF (Code=36) OR (Code=37) THEN CALL Smooth(N,Cp,Ixc(*))
7315  SUBEND
7320  !
7325  !
7330  !
7335  !
7340  SUB Smooth(Spread,SHORT Npoints,A(*))
7345  FOR J=1 TO Npoints-Spread
7350  Si=0
7355  FOR K=0 TO Spread-1
7360  I=J+K
7365  Sign=SGN(A(I))
7370  Si=Si+Sign*A(I)^2  ! SUM OF SQRS CORRECTED FOR SIGN OF ORIG. NUMBER
7375  NEXT K
7380  Sign=SGN(Si)
7385  Si=ABS(Si)
7390  A(J)=SQR(Si/Spread)*Sign
7395  NEXT J
7400  SUBEND
7405  !
7410  !
7415  !
7420  !
7425  SUB C_dmp
7430  OPTION BASE 1
7435 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
7440 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
7445 COM Files$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
7450 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
7455 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
7460 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
7465 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
7470 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
7475 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Times$[
20]
7480 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
7485  PRINTER IS 0
7490  PRINT SPA(15),"COMMAND DUMP"
7495  PRINT
7500  FOR L=1 TO Number_commands
7505  IF Command$(L)[1,2]<>"//" THEN 7520
7510  PRINT L;TAB(8);Command$(L)&Prefix$(L)&Word$(L)
```

Main Program Listing--continued.

```
7515  GOTO 7525
7520  PRINT L;TAB(11);Command$(L);TAB(33);Prefix$(L);TAB(57);Word$(L)
7525  NEXT L
7530  PRINT
7535  PRINT
7540  PRINT
7545  PRINTER IS 16
7550  CALL Putline(16,3,CHR$(131)&"SYSTEM BUSY"&CHR$(128))
7555  ASSIGN #6 TO "5420LF:F9"
7560  PRINT #6;Number_commands
7565  MAT PRINT #6;Commands$,Word$,Prefix$
7570  ASSIGN * TO #6
7575  CALL Putline(16,3,CHR$(128)&"           ")
7580  SUBEND
7585  !
7590  !
7595  !
7600  !
7605  SUB Splot(Code,A$)
7610  OPTION BASE 1
7615 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
7620 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
7625 COM Files$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
7630 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
7635 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
7640 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
7645 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
7650 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
7655 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
7660 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
7665  IF Code=30 THEN Lsf=1
7670  IF Code=31 THEN Poly=1
7675  IF Code=41 THEN Cur=1
7680  IF Code<>42 THEN Bye
7685  Region=VAL(A$)
7690  Pflg=0
7695 Bye:  !
7700  SUBEND
7705  !
7710  !
7715  !
7720  !
7725  SUB Zero_data_sets
7730  OPTION BASE 1
7735  ON ERROR GOTO Trouble
7740  SHORT X
7745  DIM Suf$(3)
7750  DATA "X", "S", "C"
7755  MAT READ Suf$
7760  FOR K=1 TO 3
7765  FOR J=1 TO 6
7770  Name$=VAL$(J)&Suf$(K)&":F8"
7775  CALL Putline(15+K,(J-1)*12,Name$)
7780  ! PURGE Name$
7785  GOTO 7795
7790 Makf:  CREATE Name$,26
7795  X=0
7800  ASSIGN #5 TO Name$
7805  PRINT #5;X,X,X
7810  ASSIGN * TO #5
7815  GOTO 7830
7820 Trouble:  IF ERRN=56 THEN GOTO Makf
7825  STOP
7830  NEXT J
```

Main Program Listing--continued.

```
7835  NEXT K
7840  GOTO Bye
7845 Bye:  !
7850  CALL Clear_line(15,4)
7855  SUBEND
7860  !
7865  !
7870  !
7875  !
7880  SUB Ren_file(Name$)
7885  OPTION BASE 1
7890  ON ERROR GOTO Trouble
7895  DIM Suf$(3)
7900  DATA "X", "S", "C"
7905  MAT READ Suf$
7910  FOR K=1 TO 3
7915  FOR J=1 TO 6
7920  Inpf$=VAL$(J)&Suf$(K)&":F8"
7925  Outf$=Name$[1,4]&VAL$(J)&Suf$(K)
7930  RENAME Inpf$ TO Outf$
7935  GOTO 7950
7940 Trouble:  IF ERRN<>54 THEN STOP
7945  PURGE Outf$
7950  NEXT J
7955  NEXT K
7960  GOTO Bye
7965 Bye:  !
7970  SUBEND
7975  !
7980  !
7985  !
7990  !
7995  SUB Del_files(Name$)
8000  OPTION BASE 1
8005  ON ERROR GOTO Trouble
8010  DIM Suf$(3)
8015  DATA "X", "S", "C"
8020  MAT READ Suf$
8025  FOR K=1 TO 3
8030  FOR J=1 TO 6
8035  File$=Name$[1,4]&VAL$(J)&Suf$(K)&":F8"
8040  PURGE File$
8045  GOTO 8060
8050 Trouble:  IF ERRN=56 THEN 8060
8055  STOP
8060  NEXT J
8065  NEXT K
8070 Bye:  !
8075  SUBEND
8080  !
8085  !
8090  !
8095  !
8100  SUB L1(Nc,A$(*),B$(*),C$(*),Bytes,Recs)
8105  Ls=0
8110  Ms=0
8115  Ns=0
8120  FOR J=1 TO Nc
8125  L=LEN(A$(J))
8130  M=LEN(B$(J))
8135  N=LEN(C$(J))
8140  IF L MOD 2=1 THEN L=L+1
8145  IF M MOD 2=1 THEN M=M+1
8150  IF N MOD 2=1 THEN N=N+1
8155  Ls=Ls+L+4
8160  Ms=Ms+M+4
8165  Ns=Ns+N+4
```

Main Program Listing—continued.

```
8170  NEXT J
8175  Bytes=Ls+Ms+Ns
8180  Recs=INT(Bytes/256+.51)
8185  Recs=Recs+1
8190  IF Recs=10 THEN Recs=11
8195  IF Recs=26 THEN Recs=27
8200  SUBEND
8205  !
8210  !
8215  !
8220  !
8225  SUB Points(N,SHORT Cur,Pflg,Pstart,Pstop,X_array(*),Y_array(*),Frc(*))
8230  OPTION BASE 1
8235  IF Cur=0 THEN Bye
8240  PRINTER IS 0
8245  PRINT LIN(2),"CURSOR ENABLED FOR INVESTIGATION OF POINTS",LIN(3)
8250  PRINTER IS 16
8255  K=0
8260  Index=1
8265  Pmin=1
8270  Q=N
8275  IF Pflg=0 THEN 8290
8280  Pmin=Pstart
8285  Q=Pstop
8290  ON KEY #15,5 GOTO Bye
8295  ON KEY #0,5 GOTO P100
8300  ON KEY #1,5 GOTO P30
8305  ON KEY #2,5 GOTO P10
8310  ON KEY #3,5 GOTO P1
8315  ON KEY #8,5 GOTO M100
8320  ON KEY #9,5 GOTO M30
8325  ON KEY #10,5 GOTO M10
8330  ON KEY #11,5 GOTO M1
8335  ON KEY #7,5 GOTO 8380
8340  ON KEY #6,5 GOSUB Dmp
8345  ON KEY #14,5 GOTO Curoff
8350  ON KEY #4,5 GOTO Advance
8355  ON KEY #12,6 GOTO 8365
8360  GOTO 8455
8365  P=0
8370  Eflag=0
8375  IF P<>1 THEN 8365
8380  PRINTER IS 0
8385  PRINT "INDEX: ";Index,"X: ";X_array(Index);"Y: ";Y_array(Index),"FREQ: ";F
rc(Index)
8390  PRINT
8395  PRINTER IS 16
8400  GOTO 8365
8405 P100:  K=100
8410 P30:  IF K=0 THEN K=30
8415 P10:  IF K=0 THEN K=10
8420 P1:  IF K=0 THEN K=1
8425 M100:  IF K=0 THEN K=-100
8430 M30:  IF K=0 THEN K=-30
8435 M10:  IF K=0 THEN K=-10
8440 M1:  IF K=0 THEN K=-1
8445  Lk=K
8450  IF Eflag<>0 THEN Advance
8455  CALL Pointr(K,Q,Pmin,Index,X_array(*),Y_array(*))
8460  GOTO 8365
8465 Advance:  !
8470  IF Eflag<>1 THEN 8500
8475  !
8480  IF Lk>0 THEN K=1
8485  IF Lk<0 THEN K=-1
8490  CALL Pointr(K,Q,Pmin,Index,X_array(*),Y_array(*))
```

Main Program Listing--continued.

```
8495  WAIT ABS(Lk)
8500  Eflag=1
8505  GOTO Advance
8510 Noadv:   Eflag=0
8515  GOTO 8365
8520 Dmp:  PRINTER IS 0
8525  IF Paper=Black THEN PRINT PAGE
8530  PRINTER IS 16
8535  DUMP GRAPHICS
8540  RETURN
8545 Curoff:  Cur=0
8550 Bye:  !
8555  SUBEND
8560  !
8565  SUB Pointr(REAL K,N,Pmin,Index,SHORT X(*),Y(*))
8570  Index=Index+K
8575  IF Index>N THEN Index=N
8580  IF Index<Pmin THEN Index=Pmin
8585  POINTER X(Index),Y(Index),2
8590  K=0
8595  SUBEND
8600  !
8605  !
8610  SUB Least(M,X(*),F(*),Eps,Maxdeg,Ndeg,Array(*),R(*))
8615  OPTION BASE 1
8620  Ib=Maxdeg+1
8625  Ib12=Maxdeg-1
8630  Ic=Ib+Ib12
8635  I011=Ic+Maxdeg
8640  I111=I011+M
8645  Rm=M
8650  Tol=Rm*Eps^2
8655  !
8660  Ndeg=0
8665  S=0
8670  Xsum=0
8675  FOR I=1 TO M
8680  S=S+1
8685  Xsum=Xsum+1*F(I)
8690  NEXT I
8695  Rn0=S
8700  !
8705  Ck=Xsum/Rn0
8710  Array(Ic)=Ck
8715  Error=0
8720  FOR I=1 TO M
8725  R(I)=Ck
8730  Error=Error+1*(Ck-F(I))^2
8735  NEXT I
8740  IF Ndeg=Maxdeg THEN L14
8745  IF Eps<0 THEN L3
8750  IF Error<=Tol THEN L14
8755  !
8760 L3:  Ndeg=1
8765  Es=Error
8770  Xsum=0
8775  FOR I=1 TO M
8780  Xsum=Xsum+1*X(I)
8785  NEXT I
8790  !
8795  Array(1)=Xsum/Rn0
8800  !
8805  S=0
8810  Xsum=0
8815  FOR I=1 TO M
8820  Array(I111+I)=X(I)-Array(1)
```

Main Program Listing--continued.

```
8825  S=S+1*Array(I111+I)^2
8830  Temp=F(I)-R(I)
8835  Xsum=Xsum+1*Array(I111+I)*Temp
8840  NEXT I
8845  Rn1=S
8850  !
8855  Ck=Xsum/Rn1
8860  Array(Ic+1)=Ck
8865  !
8870  Error=0
8875  FOR I=1 TO M
8880  R(I)=R(I)+Ck*Array(I111+I)
8885  Error=Error+1*(R(I)-F(I))^2
8890  NEXT I
8895  IF (Error>Es) AND (Eps>=0) THEN L12
8900  IF Ndeg=Maxdeg THEN L14
8905  IF (Error<=Tol) AND (Eps>=0) THEN L14
8910  FOR I=1 TO M
8915  Array(I011+I)=1
8920  NEXT I
8925  Ndeg=2
8930  K=2
8935  !
8940 L8:  Es=Error
8945  !
8950  Array(Ib12+K)=Rn1/Rn0
8955  !
8960  Xsum=0
8965  FOR I=1 TO M
8970  Xsum=Xsum+1*X(I)*Array(I111+I)^2
8975  NEXT I
8980  Array(K)=Xsum/Rn1
8985  !
8990  S=0
8995  Xsum=0
9000  FOR I=1 TO M
9005  Array(I011+I)=(X(I)-Array(K))*Array(I111+I)-Array(Ib12+K)*Array(I011+I)
9010  S=S+1*Array(I011+I)^2
9015  Temp=F(I)-R(I)
9020  Xsum=Xsum+Array(I011+1)*Temp
9025  NEXT I
9030  Rn0=Rn1
9035  Rn1=S
9040  !
9045  It=I011
9050  I011=I111
9055  I111=It
9060  !
9065  Ck=Xsum/Rn1
9070  Array(Ic+K)=Ck
9075  !
9080  Error=0
9085  FOR I=1 TO M
9090  R(I)=R(I)+Ck*Array(I111+I)
9095  Error=Error+(R(I)-F(I))^2
9100  NEXT I
9105  IF (Error>Es) AND (Eps>=0) THEN L12
9110  IF Ndeg=Maxdeg THEN L14
9115  IF (Error<=Tol) AND (Eps>=0) THEN L14
9120  Ndeg=Ndeg+1
9125  K=K+1
9130  GOTO L8
9135  !
9140 L12:  Ndeg=Ndeg-1
9145  Error=Es
9150  FOR I=1 TO M
```

Main Program Listing—continued.

```
9155  R(I)=R(I)-Ck*Array(I111+I)
9160  NEXT I
9165  !
9170  GOTO Bye
9175  !
9180 L14:  Eps=SQR(Error/Rm)
9185 Bye:  !
9190  SUBEND
9195  !
9200  !
9205  !
9210  DEF FNEval(Y,N,Array(*),Maxdeg)
9215  OPTION BASE 1
9220  Ib=Maxdeg+1
9225  Ic=Maxdeg+Ib-1
9230  !
9235  IF N>0 THEN L1
9240  RETURN Array(Ic)
9245  !
9250 L1:  IF N>1 THEN L2
9255  RETURN Array(Ic)+Array(Ic+1)*(Y-Array(1))
9260  !
9265 L2:  Dkp2=Array(Ic+N)
9270  Dkp1=Array(Ic+N-1)+(Y-Array(N))*Dkp2
9275  N12=N-2
9280  IF N12<1 THEN L4
9285  FOR L=1 TO N12
9290  K=1+N12-L
9295  Dk=Array(Ic+K)+(Y-Array(K+1))*Dkp1-Array(Ib+K)*Dkp2
9300  Dkp2=Dkp1
9305  Dkp1=Dk
9310  NEXT L
9315 L4:  RETURN Array(Ic)+(Y-Array(1))*Dkp1-Array(Ib)*Dkp2
9320 Bye:  !
9325  FNEND
9330  !
9335  !
9340  !
9345  !
9350  !
9355  !
9360  SUB Fit(Maxdeg,Desdeg,N,SHORT X_array(*),Y_array(*))
9365  OPTION BASE 1
9370 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
9375 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
9380 COM Files,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
9385 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
9390 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
9395 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
9400 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
9405 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
9410 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
9415 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
9420  DIM X(30),F(30),R(30),Array(200),Ip(30)
9425  IF Region=0 THEN 9450
9430  P1=Pstart
9435  P2=Pstop
9440  GOTO C
9445  !
9450  Index=1
9455  Pmin=1
9460  Sflg=Sflg+1
9465  IF Sflg=2 THEN Bye
```

Main Program Listing—continued.

```
9470  ON KEY #15,5 GOTO Onward
9475  ON KEY #0,5 GOTO P100
9480  ON KEY #1,5 GOTO P30
9485  ON KEY #2,5 GOTO P10
9490  ON KEY #3,5 GOTO P1
9495  ON KEY #8,5 GOTO M100
9500  ON KEY #9,5 GOTO M30
9505  ON KEY #10,5 GOTO M10
9510  ON KEY #11,5 GOTO M1
9515  ON KEY #5,5 GOTO Pnt1
9520  ON KEY #13,5 GOTO Pnt2
9525  ON KEY #6,6 GOSUB Dmp
9530  GOTO B
9535 A:  P=0
9540  IF P<>1 THEN A
9545 Pnt1:  P1=Index
9550  BEEP
9555  GOTO A
9560 Dmp:  PRINTER IS 0
9565  IF Paper=Black THEN PRINT PAGE
9570  PRINTER IS 16
9575  DUMP GRAPHICS
9580  RETURN
9585 Pnt2:  P2=Index
9590  BEEP
9595  GOTO A
9600 P100:  K=100
9605 P30:  IF K=0 THEN K=30
9610 P10:  IF K=0 THEN K=10
9615 P1:  IF K=0 THEN K=1
9620 M100:  IF K=0 THEN K=-100
9625 M30:  IF K=0 THEN K=-30
9630 M10:  IF K=0 THEN K=-10
9635 M1:  IF K=0 THEN K=-1
9640 B:  CALL Pointr(K,N,Pmin,Index,X_array(*),Y_array(*))
9645  GOTO A
9650  !
9655 Onward:  T=P2
9660  IF P2<P1 THEN P2=P1
9665  IF P2=P1 THEN P1=T
9670  !
9675 C:  !
9680  IF Maxdeg>18 THEN Maxdeg=18
9685  Sflg=1
9690  Q1=P1
9695  Q2=P2
9700  IF Region<>0 THEN Q1=Pstart
9705  IF Region<>0 THEN Q2=Pstop
9710  IF Find_point=0 THEN CALL Autoq(I,X(*),F(*),Q1,Q2,N,X_array(*),Y_array(*))
9715  IF Find_point=1 THEN CALL Autol(I,X(*),F(*),Q1,Q2,N,X_array(*),Y_array(*))
9720  IF Find_point=2 THEN CALL Manul(I,X(*),F(*),Q1,Q2,N,X_array(*),Y_array(*))
9725  FOR K=1 TO I
9730  PLOT X(K),Ymin,-2
9735  PLOT X(K),Ymin,-1
9740  PLOT X(K),Ymax,-1
9745  NEXT K
9750  M=I
9755  Eps=-1
9760  Ndeg=M-2
9765  Maxdeg=M-2
9770  IF Maxdeg>10 THEN Maxdeg=10
9775  Ndeg=Maxdeg
9780  CALL Least(M,X(*),F(*),Eps,Maxdeg,Ndeg,Array(*),R(*))
9785  ! PRINTER IS 0
9790  ! MAT PRINT F;
9795  ! PRINT
```

Main Program Listing--continued.

```
9800  ! MAT PRINT R;
9805  ! PRINT Ndeg,Eps
9810  ! PRINT
9815  Spread=X(M)-X(1)
9820  Delta=Spread/127
9825  N=M
9830  FOR J=1 TO 128
9835  Tx(J)=X(1)+Delta*(J-1)
9840  Ty(J)=FNEval(T,Ndeg,Array(*),Maxdeg)
9845  NEXT J
9850  Tnpts=128
9855 Bye:  !
9860  SUBEND
9865  !
9870  !
9875  !
9880  !
9885  SUB Swap_scale(N,SHORT X_array(*),Y_array(*))
9890  OPTION BASE 1
9895  COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
9900  COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
9905  COM Files$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np.Slope,P1,P2
9910  COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
9915  COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax
,Paxis
9920  COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,
D8,D9
9925  COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager.Nopager
9930  COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,%60,Every
9935  COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$
[20]
9940  ! COM SHORT CPX,XTP(*),CPC,CTP(*)
9945  Dnpts=N
9950  Dpscale=Pscale
9955  Dxmin=Xmin
9960  Dxmax=Xmax
9965  Dymin=Ymin
9970  Dymax=Ymax
9975  Pscale=Tpscale
9980  Xmin=Txmin
9985  Xmax=Txmax
9990  Ymin=Tymin
9995  Ymax=Tymax
10000 FOR J=1 TO Tnpts
10005 T1=X_array(J)
10010 X_array(J)=Tx(J)
10015 Tx(J)=T1
10020 T1=Y_array(J)
10025 Y_array(J)=Ty(J)
10030 Ty(J)=T1
10035 NEXT J
10040 N=Tnpts
10045 SUBEND
10050 !
10055 !
10060 SUB Unswap_scale(N,SHORT X_array(*),Y_array(*))
10065 OPTION BASE 1
10070 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
10075 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
10080 COM Files$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
10085 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
10090 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax
,Paxis
10095 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,
D8,D9
10100 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
```

Main Program Listing--continued.

```
10105 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
10110 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$
[20]
10115 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
10120 N=Dnpts
10125 Pscale=Dpscale
10130 Xmin=Dxmin
10135 Xmax=Dxmax
10140 Ymin=Dymin
10145 Ymax=Dymax
10150 FOR J=1 TO Tnpts
10155 X_array(J)=Tx(J)
10160 Y_array(J)=Ty(J)
10165 NEXT J
10170 SUBEND
10175 !
10180 !
10185 SUB Autol(Ij,X(*),F(*),P1,P2,SHORT Npts,X_array(*),Y_array(*))
10190 OPTION BASE 1
10195 DIM I(25),D(25)
10200 Ij=25
10205 P1sumx=0
10210 P2sumx=0
10215 P1sumy=0
10220 P2sumy=0
10225 S=5  ! AVE SPREAD
10230 FOR J=-S TO S
10235 P1sumx=P1sumx+X_array(P1+J-1)
10240 P1sumy=P1sumy+Y_array(P1+J-1)
10245 P2sumx=P2sumx+X_array(P2-J+1)
10250 P2sumy=P2sumy+Y_array(P2-J+1)
10255 NEXT J
10260 !
10265 I(1)=P1
10270 I(25)=P2
10275 X(1)=P1sumx/S
10280 F(1)=P1sumy/S
10285 X(25)=P2sumx/S
10290 F(25)=P2sumy/S
10295 Delt=(X(25)-X(1))/24
10300 FOR J=2 TO 24
10305 X(J)=X(1)+Delt*(J-1)
10310 NEXT J
10315 !
10320 FOR J=2 TO 24
10325 D(J)=1E99
10330 NEXT J
10335 !
10340 Xsum=0
10345 FOR J=P1 TO P1+2*S  ! AVER. FOR FIRST 2S POINTS
10350 Xsum=Xsum+X_array(J)
10355 NEXT J
10360 Ave=Xsum/(2*S)
10365 !
10370 K=P1+1
10375 FOR W=K TO P2-S  ! THE RUNNING AVERAGE
10380 Xsum=Xsum+X_array(W+S)-X_array(W-S)
10385 Ave=Xsum/(2*S)
10390 FOR J=2 TO 24
10395 IF ABS(Ave-X(J))>D(J) THEN A
10400 I(J)=W
10405 D(J)=ABS(Ave-X(J))
10410 ! IF J=2 THEN PRINT J;Diff,Minp,D(J)
10415 A: NEXT J
10420 NEXT W
10425 ! PRINTER IS 0
10430 ! MAT PRINT X;
```

Main Program Listing--continued.

```
10435 ! PRINT
10440 ! MAT PRINT I;
10445 ! PRINT
10450 ! MAT PRINT D;
10455 ! PRINT
10460 !
10465 FOR J=2 TO 24
10470 P=I(J)
10475 Sumy=0
10480 Sumx=0
10485 N=0
10490 FOR K=I(J)-S TO I(J)+S
10495 N=N+1
10500 P=K
10505 IF P<1 THEN P=1
10510 IF P>Npts THEN P=Npts
10515 Sumy=Sumy+Y_array(P)^2*SGN(Y_array(P))
10520 Sumx=Sumx+X_array(P)^2*SGN(X_array(P))
10525 NEXT K
10530 F(J)=SQR(ABS(Sumy/N))*SGN(Sumy)
10535 X(J)=SQR(ABS(Sumx/N))*SGN(Sumx)
10540 NEXT J
10545 ! MAT PRINT X;
10550 ! PRINT
10555 ! MAT PRINT F;
10560 SUBEND
10565 !
10570 !
10575 !
10580 !
10585 SUB Lsqfit(Code,N,SHORT X_array(*),Y_array(*))
10590 OPTION BASE 1
10595 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
10600 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
10605 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
10610 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
10615 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax
,Paxis
10620 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,
D8,D9
10625 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
10630 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
10635 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$
[20]
10640 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
10645 SCALE Xmin,Xmax,Ymin,Ymax
10650 Sflg=Sflg+1
10655 IF Sflg=1 THEN 10665
10660 IF Sflg=2 THEN Bye
10665 Index=1
10670 Pmin=1
10675 IF (Region=0) AND (D8=0) THEN 10700
10680 IF D8=1 THEN CALL Lsqauto(N,X_array(*),Y_array(*),Region,Pstart,Pstop,D8,P
1,P2,Ymin,Ymax)
10685 IF D8=1 THEN Onward
10690 GOSUB Dmp
10695 GOTO Onward
10700 ON KEY #15,5 GOTO Onward
10705 ON KEY #0,5 GOTO P100
10710 ON KEY #1,5 GOTO P30
10715 ON KEY #2,5 GOTO P10
10720 ON KEY #3,5 GOTO P1
10725 ON KEY #8,5 GOTO M100
10730 ON KEY #9,5 GOTO M30
10735 ON KEY #10,5 GOTO M10
10740 ON KEY #11,5 GOTO M1
```

Main Program Listing--continued.

```
10745 ON KEY #5,5 GOTO Pnt1
10750 ON KEY #13,5 GOTO Pnt2
10755 ON KEY #7,5 GOTO Prnt
10760 ON KEY #6,5 GOSUB Dmp
10765 GOTO B
10770 A: P=0
10775 Pf=0
10780 IF P<>1 THEN A
10785 !
10790 Dmp: PRINTER IS 0
10795 IF Paper=Black THEN PRINT PAGE
10800 PRINTER IS 16
10805 DUMP GRAPHICS
10810 RETURN
10815 Pnt1: P1=Index
10820 BEEP
10825 PLOT X_array(Index),Ymin,-2
10830 PLOT X_array(Index),Ymax,-1
10835 PLOT X_array(Index),Y_array(Index),-1
10840 GOTO A
10845 Pnt2: P2=Index
10850 BEEP
10855 GOTO 10825
10860 !
10865 P100: K=100
10870 P30: IF K=0 THEN K=30
10875 P10: IF K=0 THEN K=10
10880 P1: IF K=0 THEN K=1
10885 M100: IF K=0 THEN K=-100
10890 M30: IF K=0 THEN K=-30
10895 M10: IF K=0 THEN K=-10
10900 M1: IF K=0 THEN K=-1
10905 B: CALL Pointr(K,N,Pmin,Index,X_array(*),Y_array(*))
10910 GOTO A
10915 !
10920 Prnt: Pf=1
10925 Onward: IF (Region=0) AND (Pflg=0) THEN GOSUB Dmp
10930 T=P2
10935 IF Pflg=0 THEN 10950
10940 P1=Pstart
10945 P2=Pstop
10950 IF P2<P1 THEN P2=P1
10955 IF P2=P1 THEN P1=T
10960 Npts=P2-P1+1
10965 Ofset=P1
10970 !
10975 Sumx=0
10980 Sumy=0
10985 Sumxy=0
10990 Sumxsq=0
10995 Sumysq=0
11000 FOR J=P1 TO P2
11005 I=J
11010 Sumx=Sumx+X_array(I)
11015 Sumxsq=Sumxsq+X_array(I)^2
11020 Sumy=Sumy+Y_array(I)
11025 Sumxy=Sumxy+X_array(I)*Y_array(I)
11030 Sumysq=Sumysq+Y_array(I)^2
11035 NEXT J
11040 B=(Sumy*Sumxsq-Sumx*Sumxy)/(Npts*Sumxsq-Sumx^2)
11045 M=(Npts*Sumxy-Sumx*Sumy)/(Npts*Sumxsq-Sumx^2)
11050 Sumx1=0
11055 Sumy1=0
11060 Sumxy=0
11065 Sumxsq=0
11070 Sumysq=0
```

Main Program Listing--continued.

```
11075 Avex=Sumx/Npts
11080 Avey=Sumy/Npts
11085 FOR J=P1 TO P2
11090 Sumx1=X_array(J)-Avex
11095 Sumy1=Y_array(J)-Avey
11100 Sumxy=Sumxy+Sumx1*Sumy1
11105 Sumxsq=Sumxsq+Sumx1^2
11110 Sumysq=Sumysq+Sumy1^2
11115 NEXT J
11120 R=Sumxy/SQR(Sumxsq*Sumysq)
11125 Slope=M
11130 Intercept=B
11135 PRINTER IS 0
11140 PRINT LIN(6),TAB(5);"EQUATION OF LINE:  Y  = ";M;" * X  + (";B;")";LIN(2)
11145 PRINT TAB(5);"CORRELATION COEFFICIENT = ";R;LIN(6)
11150 PRINTER IS 16
11155 IF Pf=1 THEN A
11160 !
11165 X1=X_array(P1)
11170 X2=X_array(P2)
11175 Delta=(X2-X1)/127
11180 ! PRINT X1,X2,Delta
11185 FOR J=1 TO 128
11190 Tx(J)=X1+Delta*(J-1)
11195 Ty(J)=M*Tx(J)+B
11200 ! PRINT Tx(J),Ty(J)
11205 NEXT J
11210 IF Tflag<>0 THEN Sc
11215 Txmin=0
11220 Txmax=Tx(1)
11225 Tymin=Ty(1)
11230 Tymax=Ty(1)
11235 FOR J=1 TO 128
11240 Txmax=MAX(Txmax,Tx(J))
11245 Tymax=MAX(Tymax,Ty(J))
11250 Tymin=MIN(Tymin,Ty(J))
11255 NEXT J
11260 Txmax=Txmax+ABS(.1*Txmax)
11265 Tymin=Tymin-ABS(.1*Tymin)
11270 Tymax=Tymax+ABS(.1*Tymax)
11275 Tpscale=0
11280 Tnpts=128
11285 Lsf=2
11290 Sc: CALL Swap_scale(N,X_array(*),Y_array(*))
11295 Bye: !
11300 CALL Analysis(Code,N,X_array(*),Y_array(*))
11305 EXIT GRAPHICS
11310 SUBEND
11315 !
11320 !
11325 !
11330 !
11335 SUB Lsqauto(N,SHORT X(*),Y(*),Region,Pstart,Pstop,D8,P1,P2,Ymin,Ymax)
11340 Lower=1
11345 Upper=N
11350 IF Region THEN Lower=Pstart
11355 IF Region THEN Upper=Pstop
11360 U=Ymax-.15*(Ymax-Ymin)
11365 L=Ymin+.15*(Ymax-Ymin)
11370 FOR J=Upper TO Lower STEP -1
11375 P1=J
11380 IF Y(J)>L THEN N2
11385 NEXT J
11390 N2: FOR J=Lower TO Upper
11395 P2=J
11400 IF Y(J)<U THEN N3
11405 NEXT J
```

Main Program Listing--continued.

```
11410 N3: IF P1<P2 THEN N4
11415 ! P1=Lower
11420 ! P2=Upper
11425 N4: PLOT X(P1),Ymin,-2
11430 PLOT X(P1),Ymax,-1
11435 PLOT X(P2),Ymin,-2
11440 PLOT X(P2),Ymax,-1
11445 PLOT X(P2),Y(P2),-2
11450 SUBEND
11455 !
11460 SUB Autoq(I,X(*),F(*),F(*),P1,P2,N,SHORT X_array(*),Y_array(*))
11465 Npts=P2-P1+1
11470 Ofset=P1
11475 S=INT((Npts-20)/20+.5)
11480 D=10 ! +- AVERAGE
11485 IF S<10 THEN D=S-1
11490 I=0 ! NODE COUNTER
11495 S1=S+P1+1
11500 S2=P2-S
11505 FOR K=S1 TO S2 STEP S
11510 I=I+1
11515 L=0
11520 Sumx=0
11525 Sumy=0
11530 S3=K-D
11535 S4=K+D
11540 FOR J=S3 TO S4
11545 L=L+1
11550 Sumx=Sumx+X_array(J)^2*SGN(X_array(J))
11555 Sumy=Sumy+Y_array(J)^2*SGN(Y_array(J))
11560 NEXT J
11565 X(I)=SQR(ABS(Sumx/L))*SGN(Sumx)
11570 F(I)=SQR(ABS(Sumy/L))*SGN(Sumy)
11575 NEXT K
11580 SUBEND
11585 !
11590 !
11595 !
11600 !
11605 SUB Manul(I,X(*),F(*),Q1,Q2,N,SHORT X_array(*),Y_array(*))
11610 OPTION BASE 1
11615 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
11620 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
11625 COM Files$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Hp,Slope,P1,P2
11630 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Fnc(*),Cp,Intercept,Frequency
11635 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax
,Paxis
11640 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,
D8,D9
11645 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
11650 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
11655 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$
[20]
11660 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
11665 DIM P(30)
11670 I=0
11675 Pmin=Q1
11680 K=0
11685 Index=1
11690 ON KEY #15,5 GOTO Onward
11695 ON KEY #0,5 GOTO P100
11700 ON KEY #1,5 GOTO P30
11705 ON KEY #2,5 GOTO P10
11710 ON KEY #3,5 GOTO P1x
11715 ON KEY #8,5 GOTO M100
11720 ON KEY #9,5 GOTO M30
```

Main Program Listing--continued.

```
11725 ON KEY #10,5 GOTO M10
11730 ON KEY #11,5 GOTO M1
11735 ON KEY #5,5 GOTO Pnt1
11740 ! ON KEY #13,5 GOTO Pnt1
11745 ON KEY #14,5 GOTO Wipeout
11750 ON KEY #7,5 GOTO Prnt
11755 ON KEY #6,5 GOSUB Dmp
11760 GOTO B
11765 A: P=0
11770 Pf=0
11775 IF P<>1 THEN A
11780 Pnt1: IF I+1>30 THEN Too_many
11785 I=I+1
11790 P(I)=Index
11795 GOTO A
11800 Prnt: Pf=1
11805 GOTO Onward
11810 Too_many: BEEP
11815 GOTO A
11820 Dmp: PRINTER IS 0
11825 IF Paper=Black THEN PRINT PAGE
11830 PRINTER IS 16
11835 DUMP GRAPHICS
11840 RETURN
11845 P100: K=100
11850 P30:IF K=0 THEN K=30
11855 P10: IF K=0 THEN K=10
11860 P1x: IF K=0 THEN K=1
11865 M100: IF K=0 THEN K=-100
11870 M30: IF K=0 THEN K=-30
11875 M10: IF K=0 THEN K=-10
11880 M1: IF K=0 THEN K=-1
11885 B: CALL Pointr(K,Q2,Pmin,Index,X_array(*),Y_array(*))
11890 GOTO A
11895 Wipeout: I=0
11900 GOTO A
11905 Onward: !
11910 D=-10
11915 PRINTER IS 0
11920 IF Pf=1 THEN PRINT LIN(3)
11925 FOR J=1 TO I
11930 Iadr=P(J)
11935 Sumx=0
11940 Sumy=0
11945 FOR K=Iadr-D TO Iadr+D
11950 L=K
11955 IF L<1 THEN L=1
11960 IF L>N THEN L=N
11965 Sumx=X_array(L)^2*SGN(X_array(L))
11970 Sumy=Y_array(L)^2*SGN(Y_array(L))
11975 NEXT K
11980 X(J)=SQR(ABS(Sumx/I)*SGN(Sumx))
11985 F(J)=SQR(ABS(Sumy/I)*SGN(Sumy))
11990 IF Pf=1 THEN PRINT J;TAB(10);"X: ";X(J);TAB(30);"Y: ";Y_array(J)
11995 NEXT J
12000 IF Pf=1 THEN PRINT LIN(4)
12005 PRINTER IS 16
12010 IF Pf=1 THEN A
12015 GOTO Bye
12020 Bye: !
12025 SUBEND
12030 !
12035 !
12040 !
12045 !
12050 SUB Region_intrest(N,SHORT X_array(*),Y_array(*))
```

Main Program Listing--continued.

```
12055 OPTION BASE 1
12060 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
12065 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
12070 COM Files,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
12075 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
12080 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax
,Paxis
12085 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,
D8,D9
12090 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
12095 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
12100 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Times
[20]
12105 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
12110 Zot=0
12115 K=0
12120 Index=1
12125 Pmin=1
12130 SCALE Xmin,Xmax,Ymin,Ymax
12135 ON KEY #15,5 GOTO Onward
12140 ON KEY #0,15 GOTO P100
12145 ON KEY #1,5 GOTO P30
12150 ON KEY #2,5 GOTO P10
12155 ON KEY #3,5 GOTO P1x
12160 ON KEY #8,5 GOTO M100
12165 ON KEY #9,5 GOTO M30
12170 ON KEY #10,5 GOTO M10
12175 ON KEY #11,5 GOTO M1
12180 ON KEY #5,5 GOTO Pnt1
12185 ON KEY #13,5 GOTO Pnt2
12190 ON KEY #7,5 GOTO Prnt
12195 ON KEY #6,5 GOTO Dmp
12200 GOTO B
12205 A: P=0
12210 Pf=0
12215 IF P<>1 THEN A
12220 Pnt1: BEEP
12225 P1=Index
12230 PLOT X_array(Index),Ymin,-2
12235 PLOT X_array(Index),Ymax,-1
12240 PLOT X_array(Index),Y_array(Index),-2
12245 GOTO A
12250 Pnt2: BEEP
12255 P2=Index
12260 GOTO 12230
12265 P100: K=100
12270 P30: IF K=0 THEN K=30
12275 P10: IF K=0 THEN K=10
12280 P1x: IF K=0 THEN K=1
12285 M100: IF K=0 THEN K=-100
12290 M30: IF K=0 THEN K=-30
12295 M10: IF K=0 THEN K=-10
12300 M1: IF K=0 THEN K=-1
12305 B: CALL Pointr(K,N,Pmin,Index,X_array(*),Y_array(*))
12310 GOTO A
12315 Onward: Zot=1
12320 Dmp: PRINTER IS 0
12325 IF Paper=Black THEN PRINT PAGE
12330 PRINTER IS 16
12335 DUMP GRAPHICS
12340 IF Zot=1 THEN 12350
12345 GOTO A
12350 Prnt: Pf=1
12355 T=P2
12360 IF P2<P1 THEN P2=P1
12365 IF P2=P1 THEN P1=T
```

Main Program Listing—continued.

```
12370 PRINTER IS 0
12375 PRINT LIN(2),TAB(5);"REGION:  P1 = ";Frc(P1);" Hz,      P2 = ";Frc(P2);" Hz
,     DATA POINTS = ";P2-P1+1;LIN(6)
12380 PRINTER IS 16
12385 IF Zot=0 THEN GOTO A
12390 Bye: Pflg=2
12395 Pstart=P1
12400 Pstop=P2
12405 GCLEAR
12410 EXIT GRAPHICS
12415 SUBEND
12420 !
12425 !
12430 !
12435 !
12440 SUB Analysis(Code,N,SHORT X_array(*),Y_array(*))
12445 OPTION BASE 1
12450 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
12455 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
12460 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
12465 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
12470 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax
,Paxis
12475 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,
D8,D9
12480 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
12485 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
12490 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$
[20]
12495 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
12500 IF Code=12 THEN A
12505 IF Code=13 THEN B
12510 IF Code=24 THEN E
12515 IF Code=17 THEN F
12520 IF Code=14 THEN C
12525 IF Code=43 THEN D
12530 GOTO Bye
12535 A: Lower=1    !  IMAGINARY vs REAL ANALYSIS
12540 IF (Region=1) OR (Poly=1) THEN Lower=Pstart
12545 Upper=N
12550 IF (Region=1) OR (Poly=1) THEN Upper=Pstop
12555 Mi=Y_array(1)
12560 Mr=X_array(1)
12565 F=Frc(1)
12570 FOR J=Lower TO Upper
12575 Mr=MIN(Mr,X_array(J))
12580 T=MIN(Mi,Y_array(J))
12585 IF T>=Mi THEN Ad
12590 F=Frc(J)
12595 Mi=T
12600 Ad: NEXT J
12605 Sz(1)=Mi
12610 Sz(2)=F
12615 Sz(3)=Mr
12620 Sz(4)=-2*Sz(1)
12625 Sz(5)=1/(2*PI*Sz(4)*Sz(2))
12630 GOTO Bye
12635 !
12640 B: Sz(8)=Slope   !  REAL vs IMAGINARY*HZ ANALYSIS
12645 Sz(9)=Intercept
12650 GOTO Bye
12655 !
12660 C: Sz(12)=Slope  !  REAL vs IMAGINARY/HZ ANALYSIS
12665 Sz(13)=Intercept
12670 GOTO Bye
12675 !
```

Main Program Listing--continued.

```
13005 PRINT LIN(4)
13010 PRINTER IS 16
13015 Bye:!
13020 SUBEND
13025 !
13030 !
13035 SUB Dummya1
13040 SUBEND
13045 !
13050 !
13055 !
13060 SUB Remove(Code,SHORT D1,Frequency,Every,X60,Cp,Ryc(*),Ixc(*),Frc(*))
13065 OPTION BASE 1
13070 IF D1<>0 THEN Bye
13075 IF Code=44 THEN Stay
13080 IF Frequency=X60 THEN Stay
13085 GOTO Bye
13090 Stay: !
13095 SHORT Bad(30),Chnls(30)
13100 SHORT Tmin,Temp,Badmax
13105 Freqency_table: DATA 21.774,10,  58,8,  117,7,  177,7, 238,5,  287.5,3
13110 DATA 350,3,  412.5,3,  475,3,  525,3,  587.5,3,  650,3,  712.5,3
13115 DATA 775,3,  825,3,  887.5,3,  950,3,  -1,-1
13120 FOR J=1 TO 30
13125 READ Bad(J),Chnls(J)
13130 IF Bad(J)<>-1 THEN 13155
13135 Bad(J)=0
13140 Chnls(J)=0
13145 Npairs=J-1
13150 GOTO 13175
13155 NEXT J
13160 !
13165 ! ARRANGE THE BAD FREQUENCIES IN ASCENDING ORDER
13170 !
13175 FOR J=1 TO Npairs
13180 Tmin=Bad(J)
13185 FOR K=J TO N
13190 Temp=MIN(Tmin,Bad(K))
13195 IF Temp>Tmin THEN A
13200 Tmin=Temp
13205 I=K
13210 A: NEXT K
13215 IF Bad(J)=Tmin THEN 13250
13220 T=Bad(J)
13225 Bad(J)=Bad(I)
13230 Bad(I)=T
13235 T=Chnls(J)
13240 Chnls(J)=Chnls(I)
13245 Chnls(I)=T
13250 NEXT J
13255 !
13260 ! NOW REMOVE FREQUENCIES FROM INCOMING DATA
13265 !
13270 B=0
13275 N=0
13280 K=1
13285 FOR J=1 TO Cp
13290 Ij=J+1
13295 N=N+1
13300 IF K>Npairs THEN B
13305 IF N>Cp THEN Out
13310 IF Frc(N)<Bad(K) THEN B
13315 N=N+Chnls(K)
13320 IF N>Cp THEN Out
13325 Ryc(J)=Ryc(N)
13330 Ixc(J)=Ixc(N)
```

Main Program Listing--continued.

```
13335 Frc(J)=Frc(N)
13340 K=K+1
13345 GOTO Next1
13350 B:!
13355 IF N>Cp THEN Out
13360 Ryc(J)=Ryc(N)
13365 Ixc(J)=Ixc(N)
13370 Frc(J)=Frc(N)
13375 Next1: IF N=Cp THEN Out
13380 NEXT J
13385 Out: FOR J=Ij TO Cp
13390 Frc(J)=Ryc(J)=Ixc(J)=0
13395 NEXT J
13400 Cp=Ij-1
13405 D1=1
13410 Bye:!
13415 SUBEND
13420 ! SUBROUTINE TO CREATE COMMAND FILES
13425 ! CALL:     CALL COMMAND_FILE
13430 !
13435 !
13440 !
13445 SUB Command_file
13450 OPTION BASE 1
13455 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
13456 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
13457 COM Files,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
13458 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
13459 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax
,Paxis
13460 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,
D8,D9
13461 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
13462 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
13463 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$
[20]
13464 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
13485 ON ERROR GOTO Trouble
13490 CALL Clear
13495 CALL Putline(0,26,"COMMAND FILE MENU")
13500 Menu=6
13505 CALL Putline(2,5,"DO YOU WISH TO SAVE, RECALL OR APPEND COMMAND FILE (S, R
, A) *")
13510 LINPUT Ans$
13515 IF Ans$="S" THEN Save_file
13520 IF Ans$="R" THEN Recall_file
13525 IF Ans$="A" THEN Append_file
13530 IF Ans$="0" THEN 13955
13535 BEEP
13540 GOTO 13490
13545 !
13550 ! SAVE A COMMAND STACK
13555 !
13560 ! CREATE COMMAND FILE
13565 Save_file:!
13570 CALL Putline(4,5,"WHAT IS THE FILE TO BE NAMED ?")
13575 INPUT Name$
13580 ! IF NO ":" IN NAME$, APPEND ":F9"
13585 P=POS(Name$,":")
13590 IF P=0 THEN Name$=Name$&":F9"
13595 F=1
13600 CALL L1(Number_commands,Command$(*),Prefix$(*),Word$(*),Bytes,Recs)
13605 PRINT "NUMBER OF BYTES: ";Bytes,"NUMBER OF RECORDS: ";Recs
13610 CREATE Name$,Recs
13615 ASSIGN #9 TO Name$
13620 PRINT #9;Number_commands
```

Main Program Listing—concluded.

```
13625 FOR J=1 TO Number_commands
13630 PRINT #9;Command$(J),Word$(J),Prefix$(J)
```

Diskette Subroutine Listing

As mentioned earlier, the activation of certain function keys caused a subroutine to be read in from the diskette and executed. Table D-2 lists the subroutine associated with each of those keys. The unlisted keys activate subroutines in the main program. Following the table is a listing of each of the individual subroutines.

TABLE D-2 KEY ACTIVATED DISKETTE SUBROUTINES

KEY #	SUBROUTINE
0	Setup_group
1	Cursor_group
2	Display_group
3	Control_group
4	Special_group
6	Command_file
7	Dfile
8	Dmanip
11	Mul_file
12	Com_group
14	Plot_group

Subroutine "Setup __ group"

```
10   ! SUB Setup_group
20   ! SUBROUTINE TO DISPLAY SETUP GROUP MENU
30   ! CALL:     CALL Setup_group
40   !
50   SUB Setup_group
60   OPTION BASE 1
70  COM Command$(*),Humber_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
80  COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
90  COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
100 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
110 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,P
axis
120 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D8
,D9
130 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
140 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
150 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[2
0]
160 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
170  DIM Setup_command$(18),Set$(8),Reset$(8),English$(18)
180  !
190  ! THESE ARE THE HPIB BUS COMMAND CODES
200  !
210  DATA " TI"," FR"," AV"," SG"," TG"," CH"
220  DATA " RG"," AC"," DE"," CA"," DN"," UP"
230  DATA " CF"," BW"," TL"," SO"," VW"," RO"
240  MAT READ Setup_command$
250  !
260  ! THESE ARE THE SET$ & RESET ARRAYS
270  ! WHAT IS IN SET$ IS CHANGED TO WHAT IS IN RESET$
280  !
290   DATA "KHZ","VOLT","MSEC","DB","MHZ","USEC","SEC","HZ"
300  DATA  "KV ","KV  ","HD  ","HD","MU ","MU  ","KV ","HD"
310  MAT READ Set$
320  MAT READ Reset$
330  !
340  DATA "TIME","FREQ","AVG","SGNL","TRIG","CHAN #","RANGE","AC,DC","DELAY"
350  DATA "CAL","DN-ARROW","UP-ARROW","CENT FREQ","BW","T","STORE","VIEW"
360  DATA "RESTORE"
370  ! FULL WORD OF CHOICE
380  MAT READ English$
390  !
400  ! PUT MENU ON SCREEN
410  ! READ FULL WORDS
420  !
430  CALL Clear
440  IF Editf$<>" " THEN CALL Putline(0,0,CHR$(131)&"EDIT MODE"&CHR$(128))
450  CALL Putline(0,26,"SETUP GROUP MENU")
460  CALL Group_menu(2,7,3,10,20,18,English$(*))
470  CALL Putline(9,0,"INPUT YOUR CHOICE *")
480  Menu=3
490  !
500  ! GET INPUT CHOICE NUMBER
510  !
520  INPUT Setup_choice
530  IF Setup_choice=0 THEN 940  ! IF=0 THEN EXIT
540  IF Setup_choice<0 THEN Setup_err
550  IF Setup_choice>18 THEN Setup_err
560  GOTO 610
570 Setup_err: !
580  BEEP
590  PRINT "IMPROPER CHOICE -- TRY AGAIN"
600  GOTO 470
610  CALL Get_p(Parm$,L)
620   IF (Editf$="ADD") OR (Editf$="REP") THEN L=Edit_line
630  IF Editf$="ADD" THEN CALL Shifty(1)
```

Subroutine "Setup_ group" continued.

```
640  !
650  !
660  ! CHECK PARAMETER STRING FOR ANY STRING IN SET$
670  ! AND CHANGE IT TO WHAT IS IN RESET$
680  !
690  Prefix$(L)=Parm$
700  FOR J=1 TO 8
710  P=POS(Parm$,Set$(J))
720  IF P=0 THEN 770
730  Q=LEN(Set$(J))-1
740  ! PRINT P,Q
750  Parm$[P,P+Q]=Reset$(J)
760  GOTO 710
770  NEXT J
780  !
790  ! CONCATENATE STRINGS AND PUT ";" ON END
800  !
810  Command$(L)=Parm$&Setup_command$(Setup_choice)&";"
820  Word$(L)=English$(Setup_choice)
830  !
840  ! BLANK OUT OLD LINES ON SCREEN
850  !
860  CALL Clear_line(11,8)
870  PRINT Command$(L);"                        "
880  IF Mode$="IMM" THEN CALL Send
890  IF Editf$="REP" THEN 940
900  !
910  ! GO GET MORE COMMANDS OF THIS GROUP
920  !
930  GOTO 470
940  SUBEND
```

Subroutine "Cursor _ group"

```
10   ! SUBROUTINE TO DISPLAY CURSOR GROUP MENU
20   !
30   ! CALL:     CALL CURSOR_GROUP
40   !
50   SUB Cursor_group
60   OPTION BASE 1
70   COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
80   COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
90   COM Files,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
100  COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
110  COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
120  COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
130  COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
140  COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
150  COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
160  ! COM SHORT CPX,XTP(*),CPC,CTP(*)
170  CALL Clear
180  Menu=1
190  IF Editf$<>" " THEN CALL Putline(0,0,CHR$(131)&"EDIT MODE"&CHR$(128))
200  CALL Putline(2,13,"CURSOR CANNOT BE CONTROLLED BY THE COMPUTER")
210  CALL Putline(4,19,"USE 5420 FRONT PANEL CONTROLS")
220  WAIT 3000
230  SUBEND
```

Subroutine "Display _ group"

```
1110 !
1115 ! SUBROUTINE FOR DISPLAY GROUP
1120 ! CALL:     CALL DISPLAY_GROUP
1125 SUB Display_group
1130 OPTION BASE 1
1135 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
1136 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
1137 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
1138 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
1139 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
1140 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
1141 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
1142 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
1143 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
1144 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
1165 DIM Display_command$(30),English$(30)
1170 DATA " MG"," PH"," LM"," LX"," RE"," IM"," IR"," LP"
1175 DATA " TC"," FM"," WT"," EX"," AD"," SB"," MY"," DV"
1180 DATA " HM"," PK"," JW"," PW"," PR"," PL"," RA"," SA"
1185 !
1190 ! HPIB COMMANDS FOR 5420
1195 !
1200 DATA " TN"," CO"," AU"," CR"," HI"," LI"
1205 MAT READ Display_command$
1210 DATA "MAG","PHASE","LOG MAG","X, LOG X","REAL","IMAG","IM/RE","LG MG/<"
1215 DATA "TRACE","FORMAT","WEIGHT","EXPAND","+","-","*","/","HMNC","PEAK","JW"
1220 DATA "POWER","PRINT","PLOT","RECALL","SAVE","TRANS","COHER","AUTO","CROSS"
1225 !
1230 ! FULL WORD DESCRIPTIONS
1235 !
1240 DATA "HIST","LINEAR"
1245 MAT READ English$
1250 CALL Clear
1255 IF Editf$<>" " THEN CALL Putline(0,0,CHR$(131)&"EDIT MODE"&CHR$(128))
1260 CALL Putline(0,29,"DISPLAY GROUP MENU")
1265 CALL Group_menu(2,7,5,2,15,30,English$(*))
1270 CALL Putline(9,0,"INPUT YOUR CHOICE *")
1275 Menu=2
1280 INPUT Display_choice
1285 IF Display_choice=0 THEN 1425
1290 IF Display_choice<1 THEN Display_err
1295 !
1300 ! GET USER CHOICE
1305 !
1310 IF Display_choice>30 THEN Display_err
1315 ! GET PARAMETER INPUT
1320 !
1325 GOTO 1350
1330 Display_err: !
1335 BEEP
1340 PRINT "IMPROPER CHOICE -- TRY AGAIN"
1345 GOTO 1270
1350 CALL Get_p(Parm$,L)
1355 Build_command: !
1360  IF (Editf$="ADD") OR (Editf$="REP") THEN L=Edit_line
1365  IF Editf$="ADD" THEN CALL Shifty(1)
1370 Prefix$(L)=Parm$
1375 !
1380 ! REMOVE QUOTE MARKS
1385 !
1390 Word$(L)=English$(Display_choice)
1395 Command$(L)=Parm$&Display_command$(Display_choice)&";"
1400 CALL Clear_line(11,0)
```

Subroutine "Display __ group" continued

```
1405 PRINT Command$(L);"
1410 IF Mode$="IMM" THEN CALL Send
1415 IF Editf$="REP" THEN 1425
1420 GOTO 1270
1425 SUBEND
```

Subroutine "Control _ group"

```
1460 ! SUBROUTINE FOR CONTROL GROUP
1465 ! CALL:     CALL CONTROL_GROUP
1470 !
1475 !
1480 SUB Control_group
1485 OPTION BASE 1
1490 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
1491 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
1492 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
1493 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
1494 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
1495 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
1496 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
1497 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
1498 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
1499 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
1520 DIM Control_command$(6),English$(6)
1525 ! HPIB COMMANDS FOR CONTROL FUNCTIONS
1530 DATA " ST"," PA"," VI"," MR"," RS"," SL"
1535 MAT READ Control_command$
1540 DATA "START","PAUSE/CONT","VIEW INPUT","MAX RATE","RESET","SELF TEST"
1545 MAT READ English$
1550 CALL Clear
1555 IF Editf$<>" " THEN CALL Putline(0,0,CHR$(131)&"EDIT MODE"&CHR$(128))
1560 CALL Putline(0,26,"CONTROL GROUP MENU")
1565 CALL Group_menu(2,3,3,10,20,6,English$(*))
1570 Menu=3
1575 Query: !
1580 CALL Putline(6,0,"INPUT YOUR CHOICE *")
1585 ! GET CONTROL CHOICE
1590 INPUT Control_choice
1595 IF Control_choice=0 THEN Bye
1600 IF Control_choice<1 THEN Control_err
1605 IF Control_choice>6 THEN Control_err
1610 GOTO Inc
1615 Control_err: !
1620 BEEP
1625 PRINT "IMPROPER CHOICE -- TRY AGAIN"
1630 GOTO 1590
1635 ! ASSEMBLE COMMAND
1640 Inc: !
1645 CALL Get_p(Parm$,L)
1650 Build: !
1655  IF (Editf$="ADD") OR (Editf$="REP") THEN L=Edit_line
1660  IF Editf$="ADD" THEN CALL Shifty(1)
1665 Prefix$(L)=Parm$
1670 Command$(L)=Parm$&Control_command$(Control_choice)&";"
1675 Word$(L)=English$(Control_choice)
1680 ! CLEAR SCREEN LINES
1685 CALL Clear_line(0,8)
1690 PRINT Command$(L);"
1695 IF Mode$="IMM" THEN CALL Send
1700 IF Editf$="REP" THEN Bye
1705 GOTO Query
1710 Bye: !
1715 SUBEND
```

Subroutine "Special _ group"

```
10   ! SUBROUTINE FOR SPECIAL_GROUP
20   ! CALL:     CALL SPECIAL_GROUP
30   !
40   !
50   !
60   SUB Special_group
70   OPTION BASE 1
80   COM Commands$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
90   COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
100  COM Files$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
110  COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
120  COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
130  COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5.D6,D7,D
8,D9
140  COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
150  COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
160  COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
170  ! COM SHORT CPX,XTP(*),CPC,CTP(*)
180  DIM Special_commands$(8),English$(9),A$[51]
190  ! HPIB COMMANDS FOR SPECIAL COMMANDS
200  DATA " TX"," OA"," LL"," GL"," DI"," EN"," Device Clear"," Serial Poll"
210  MAT READ Special_commands$
220  DATA "WRITE TEXT","# AVERAGES","LOC LOCKOUT","GOTO LOCAL"
230  DATA "DISABLE SRQ","ENABLE SRQ","DEVICE CLR","SERIAL POLL","COMMENT"
240  MAT READ English$
250  ! PUT MENU ON CRT
260  CALL Clear
270  IF Editf$<>" " THEN CALL Putline(0,0,CHR$(131)&"EDIT MODE"&CHR$(128))
280  CALL Putline(0,30,"SPECIAL GROUP MENU")
290  CALL Group_menu(2,4,3,5,25,9,English$(*))
300  Menu=4
310 Query:  !
320  CALL Putline(0,0,"INPUT YOUR CHOICE #")
330  ! INPUT MENU CHOICE
340  INPUT Special_choice
350  IF Special_choice=0 THEN Bye
360  IF Special_choice<0 THEN Special_err
370  IF Special_choice>9 THEN Special_err
380   IF Special_choice=9 THEN Build
390  IF Special_choice<>10 THEN Inc
400  !
410 Special_err:  !
420  BEEP
430  PRINT "IMPROPER CHOICE -- TRY AGAIN"
440  GOTO Query
450  ! BUILD COMMANDS
460 Inc:  !
470  CALL Get_p(Parm$,L)
480  IF Special_choice<>1 THEN Build
490  PRINT "INPUT TEXT STRING (NO COMMAS ALLOWED) OR ELSE / "
500  LINPUT Suffix$
510  P=POS(Suffix$,"/")
520  IF P<>0 THEN Suffix$=" "
530  ! BUILD COMMAND STRING
540 Build:  !
550   IF (Editf$="REP") OR (Editf$="ADD") THEN L=Edit_line
560   IF Editf$="ADD" THEN CALL Shifty(1)
570   IF Special_choice<>9 THEN 800
580   Number_commands=Number_commands+1
581   L=Number_commands
590   IF Editf$="REP" THEN L=Edit_line
591   IF Editf$="ADD" THEN L=Edit_line-1
592   A$="...................................................."
593   PRINT A$
600   LINPUT A$
```

Subroutine "Special _ group" continued

```
610   A$=TRIM$(A$)
620   Ln=LEN(A$)
630   Word$(L)=" "
640   Prefix$(L)=" "
650   Command$(L)=" "
660   Q=15
670   IF Ln<15 THEN Q=Ln
680   Command$(L)="// "&A$[1,Q]
690   Q=16
700   Z=33
710   IF Ln<16 THEN Out
720   IF Ln<Z THEN Z=Ln
730   Prefix$(L)=A$[Q,Z]
740   IF Ln<34 THEN Out
750   Q=34
760   Z=51
770   IF Ln<Z THEN Z=Ln
780   Word$(L)=A$[Q,Z]
790 Out:   GOTO 900
800  Command$(L)=Parm$&Special_command$(Special_choice)&";"
810  IF Special_choice<>1 THEN GOTO 870
820  P=POS(Command$(L),";")
830  X$=Command$(L)
840  X$[P]=" "
850  X$=X$&Suffix$&";"
860  Command$(L)=X$
870  Word$(L)=English$(Special_choice)
880  Prefix$(L)=Parm$
890  ! CLEAR SCREEN LINES
900  CALL Clear_line(8,11)
910  PRINT Command$(L);"                    "
920  IF Mode$="IMM" THEN CALL Send
930  IF Editf$="REP" THEN Bye
940  GOTO Query
950 Bye:  !
960  SUBEND
```

Subroutine "Command_file"

```
2605 ! SUBROUTINE TO CREATE COMMAND FILES
2610 ! CALL:     CALL COMMAND_FILE
2615 !
2620 !
2625 !
2630 SUB Command_file
2635 OPTION BASE 1
2640 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
2641 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
2642 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
2643 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
2644 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
2645 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
2646 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
2647 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
2648 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
2649 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
2670  ON ERROR GOTO Trouble
2675 CALL Clear
2680 CALL Putline(0,26,"COMMAND FILE MENU")
2685 Menu=6
2690 CALL Putline(2,5,"DO YOU WISH TO SAVE, RECALL OR APPEND COMMAND FILE (S, R,
 A) *")
2695 LINPUT Ans$
2700 IF Ans$="S" THEN Save_file
2705 IF Ans$="R" THEN Recall_file
2710  IF Ans$="A" THEN Append_file
2715  IF Ans$="0" THEN 3140
2720 BEEP
2725 GOTO 2675
2730 !
2735 ! SAVE A COMMAND STACK
2740 !
2745 ! CREATE COMMAND FILE
2750 Save_file: !
2755 CALL Putline(4,5,"WHAT IS THE FILE TO BE NAMED ?")
2760 INPUT Name$
2765 ! IF NO ":" IN NAME$, APPEND ":F9"
2770 P=POS(Name$,":")
2775 IF P=0 THEN Name$=Name$&":F9"
2780  F=1
2785  CALL L1(Number_commands,Command$(*),Prefix$(*),Word$(*),Bytes,Recs)
2790  PRINT "NUMBER OF BYTES: ";Bytes,"NUMBER OF RECORDS: ";Recs
2795  CREATE Name$,Recs
2800 ASSIGN #9 TO Name$
2805 PRINT #9;Number_commands
2810  FOR J=1 TO Number_commands
2815  PRINT #9;Command$(J),Word$(J),Prefix$(J)
2820  NEXT J
2825 ASSIGN * TO #9
2830 Menu=6
2835  CALL Clear_line(12,1)
2840 GOTO 3140
2845 !
2850 ! RECALL AN OLD COMMAND FILE
2855 !
2860 Recall_file: !
2865 CALL Putline(4,5,"WHAT IS THE FILENAME ?")
2870 LINPUT Name$
2875  IF POS(Name$,":")=0 THEN Name$=Name$&":F9"
2880 File$=Name$
2885  F=2
2890 ASSIGN #8 TO Name$
2895 READ #8;Number_commands
```

Subroutine "Command _ file" continued

```
2900 ! MAT READ #8;Command$,Word$,Prefix$
2905  FOR J=1 TO Number_commands
2910  READ #8;Command$(J),Word$(J),Prefix$(J)
2915  NEXT J
2920 ASSIGN * TO #8
2925  CALL Clear_line(12,1)
2930 Menu=6
2935 GOTO 3140
2940 !
2945 ! APPEND A FILE
2950 Append_file: !
2955  F=3
2960 CALL Putline(4,5,"WHAT IS THE FILE NAME ?")
2965 INPUT Name$
2970  IF POS(Name$,":")=0 THEN Name$=Name$&":F9"
2975 ASSIGN #7 TO Name$
2980  BUFFER #7
2985 READ #7;Number
2990  FOR J=1 TO Number
2995  I=Number_commands+J
3000  READ #7;Command$(I),Word$(I),Prefix$(I)
3005 NEXT J
3010 ASSIGN * TO #7
3015 Menu=6
3020 Number_commands=Number_commands+Number
3025  CALL Clear_line(12,1)
3030  GOTO 3140
3035 Trouble:  !
3040  IF ERRN=56 THEN No_file
3045  IF ERRN=54 THEN Dup_file
3050  STOP
3055 No_file:  !
3060  BEEP
3065  PRINT
3070  PRINT "FILE NON-EXSISTANT --  TRY AGAIN"
3075  WAIT 2000
3080  CALL Clear_line(4,12)
3085  IF F=2 THEN Recall_file
3090  IF F=3 THEN Append_file
3095  STOP
3100 Dup_file:  !
3105  BEEP
3110  PRINT
3115  PRINT "FILE ALREADY EXISTS -- TRY AGAIN"
3120  WAIT 2000
3125  CALL Clear_line(4,12)
3130  IF F=1 THEN Save_file
3135  STOP
3140 SUBEND
```

Subroutine "Dfile"

```
7710  ! SUBROUTINE TO ASK PROMPTS FOR DATA FILE NAME
7715  ! CALL:     CALL DFILE
7720  !
7725  SUB Dfile
7730  OPTION BASE 1
7735 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
7736 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
7737 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
7738 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
7739 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
7740 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
7741 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
7742 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
7743 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
7744 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
7765  CALL Clear
7770  IF Editf$<>" " THEN CALL Putline(0,0,CHR$(131)&"EDIT MODE"&CHR$(128))
7775  CALL Putline(0,26,"DATA FILE MENU")
7780  CALL Putline(2,5,"1  =  SAVE SHORT DATA")
7785  CALL Putline(2,35,"2  =  RECALL SHORT DATA")
7790  CALL Putline(3,5,"3  =  SAVE TIED DATA")
7795  CALL Putline(3,35,"4  =  RECALL TIED DATA")
7800  CALL Putline(4,5,"5  =  CREATE ZEROED SETS")
7805  CALL Putline(4,35,"6  =  RENAME DATA SETS")
7810  CALL Putline(5,5,"7  =  DELETE DATA SETS")
7815  CALL Putline(7,5,"INPUT YOUR CHOICE*")
7820  INPUT Dchoice
7825  IF Dchoice=0 THEN Bye
7830  Number_commands=Number_commands+1
7835  IF (Dchoice<1) OR (Dchoice>7) THEN 7765
7840  IF Dchoice=1 THEN Dchoice=9
7845  IF Dchoice=2 THEN Dchoice=10
7850  IF Dchoice=3 THEN Dchoice=18
7855  IF Dchoice=4 THEN Dchoice=19
7860  IF Dchoice=5 THEN Dchoice=20
7865  IF Dchoice=6 THEN Dchoice=21
7870  IF Dchoice=7 THEN Dchoice=39
7875  IF Dchoice=20 THEN Zeroed
7880  IF Dchoice=21 THEN Ren_file
7885  IF Dchoice=39 THEN Del_fil
7890  CALL Putline(9,4,"INPUT FILE-NAME:")
7895  LINPUT Name$
7900  L=Number_commands
7905  IF (Editf$="ADD") OR (Editf$="REP") THEN L=Edit_line
7910  IF Editf$="ADD" THEN CALL Shifty(1)
7915  Command$(L)="*SYS*"
7920  Prefix$(L)=VAL$(Dchoice)
7925  Word$(L)=Name$
7930  ! IF (Editf$=" ") OR (Editf$="ADD") THEN Number_commands=Number_commands+1
7935  CALL Clear_line(6,10)
7940  GOTO 7815
7945 Zeroed:  !
7950  Name$="ZERO DATA SETS"
7955  GOTO 7900
7960 Ren_file:  PRINT "INPUT DESIRED NAME (4 CHARS MAX) :"
7965  INPUT Name$
7970  GOTO 7900
7975 Del_fil:  PRINT "INPUT DESIRED NAME (4 CHARS MAX) :"
7980  INPUT Name$
7985  GOTO 7900
7990 Bye:  !
7995  Menu=7
8000  SUBEND
```

Subroutine "Dmanip"

```
10    SUB Dmanip
20    OPTION BASE 1
30   COM Commands$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
40   COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
50   COM Files,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
60   COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
70   COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
80   COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
90   COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
100  COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
110  COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
120  ! COM SHORT CPX,XTP(*),CPC,CTP(*)
130   CALL Clear
140   IF Editf$<>" " THEN CALL Putline(0,0,CHR$(131)&"EDIT MODE"&CHR$(128))
150   CALL Putline(0,26,"DATA MANIPULATIONS GROUP")
160   CALL Putline(2,5,"1 = TIE DATA SETS")
170   CALL Putline(2,35,"2 = SWAP REAL & IMAG")
180   CALL Putline(3,5,"3 = SCALE REAL & IMAG")
190   CALL Putline(3,35,"4 = LEAST SQ. FIT")
200   CALL Putline(4,5,"5 = POLYNOMIAL FIT")
210   CALL Putline(4,35,"6 = SMOOTH REAL SHORT")
220   CALL Putline(5,5,"7 = SMOOTH IMAG SHORT")
230   CALL Putline(5,35,"8 = SMOOTH BOTH SHORT")
240   CALL Putline(6,5,"9 = SMOOTH REAL TIED")
250   CALL Putline(6,35,"10= SMOOTH IMAG TIED")
260   CALL Putline(7,5,"11= SMOOTH BOTH TIED")
270   CALL Putline(7,35,"12= MOVE SHORT TO TIED")
280   CALL Putline(8,5,"13= CURSOR")
290   CALL Putline(8,35,"14= REGION OF INTEREST")
300   CALL Putline(9,5,"INPUT YOUR CHOICE *")
310   INPUT Dchoice
320   IF Dchoice=0 THEN Bye
330   IF Dchoice=1 THEN Tie
340   IF (Dchoice>=2) AND (Dchoice<=14) THEN Manip
350   IF (Dchoice<0) OR (Dchoice>14) THEN 130
360 Tie:   ! NC
370   CALL Putline(11,5,"INPUT FILES:")
380   LINPUT F$
390   IF (Editf$=" ") OR (Editf$="ADD") THEN Number_commands=Number_commands+1
400   L=Number_commands
410   IF (Editf$="REP") OR (Editf$="ADD") THEN L=Edit_line
420   IF Editf$="ADD" THEN CALL Shifty(1)
430   Commands$(L)="*SYS*"
440   Prefix$(L)=VAL$(11)
450   Word$(L)=F$
460   CALL Clear_line(7,10)
470   GOTO 310
480 Manip:   !
490   IF (Editf$=" ") OR (Editf$="ADD") THEN Number_commands=Number_commands+1
500   L=Number_commands
510   IF (Editf$="ADD") OR (Editf$="REP") THEN L=Edit_line
520   IF Editf$="ADD" THEN CALL Shifty(1)
530   Commands$(L)="*SYS*"
540   IF Dchoice=2 THEN Prefix$(L)=VAL$(22)
550   IF Dchoice=3 THEN Prefix$(L)=VAL$(23)
560   IF (Dchoice>=4) AND (Dchoice<=12) THEN Prefix$(L)=VAL$(Dchoice+26)
570   IF (Dchoice>=13) AND (Dchoice<=14) THEN Prefix$(L)=VAL$(Dchoice+28)
580   IF Dchoice=2 THEN Word$(L)="SWAP REAL,IMAG"
590   IF Dchoice=3 THEN CALL Putline(9,5,"INPUT FUNCTION (* /)")
600   IF Dchoice=3 THEN LINPUT Func$
610   IF Dchoice=3 THEN Fac$=" "
620   IF Dchoice=3 THEN PRINT "INPUT VALUE FOR SCALING"
630   IF Dchoice=3 THEN INPUT Fac$
```

Subroutine "Dmanip" continued

```
640   IF Dchoice=3 THEN PRINT "REAL OR IMAGINARY (R or I or RI) ? "
650   IF Dchoice=3 THEN INPUT Rib$
651   IF (Dchoice=3) AND (LEN(Rib$)=0) THEN Rib$="RI"
660   IF Dchoice=3 THEN Rib$=TRIM$(Rib$)
670   IF Dchoice=3 THEN Word$(L)=TRIM$(Func$&","&Fac$&","&Rib$)
680   IF Dchoice=4 THEN Word$(L)="LEAST SQ."
690   IF (Dchoice>=6) AND (Dchoice<=11) THEN PRINT "INPUT TIMES SMOOTHED"
700   IF (Dchoice>=6) AND (Dchoice<=11) THEN INPUT Times
710   IF (Dchoice>=6) AND (Dchoice<=11) THEN Word$(L)=VAL$(Times)
720   IF Dchoice=12 THEN Word$(L)="SHORT TO TIED"
730   IF Dchoice=14 THEN PRINT "REGION OF INTREST: (1=ON   0=OFF) ?"
740   IF Dchoice=14 THEN INPUT Word$(L)
750   IF Dchoice=13 THEN Word$(L)="CURSOR"
760   IF Dchoice=5 THEN PRINT "POLYNOMIAL POINT FINDER: (1=QUICK  2=LONG  3=MANU
AL) ?"
770   IF Dchoice=5 THEN INPUT Word$(L)
780   CALL Clear_line(9,11)
790   GOTO 300
800 Bye:   !
810   Menu=0
820   SUBEND
```

Subroutine "Mul__file"

```
4655 ! SUBROUTINE FOR MULTIPLE FILES
4660 ! CALL:     CALL MUL_FILE
4665 SUB Mul_file
4670 OPTION BASE 1
4675 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
4676 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
4677 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
4678 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
4679 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
4680 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
4681 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
4682 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
4683 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
4684 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
4705  ON ERROR GOTO Trouble
4710 Menu=11
4715 CALL Clear
4720 CALL Putline(0,24,"FILE-OF-FILES MENU")
4725 CALL Putline(2,0,"YOU MAY BUILD, EXECUTE OR LIST A FILE-OF-FILES:")
4730 CALL Putline(4,5,"INPUT YOUR CHOICE (B E L) *")
4735 LINPUT Choice$
4740 IF Choice$="0" THEN Bye
4745 CALL Putline(6,0,"INPUT FILE NAME:")
4750 LINPUT Name$
4755  IF POS(Name$,":")=0 THEN Name$=Name$&":F9"
4760 IF Choice$="B" THEN Build
4765 IF Choice$="E" THEN Execute
4770 IF Choice$="L" THEN Listc
4775 BEEP
4780 GOTO 4715
4785 Build: !
4790 CREATE Name$,10
4795 ASSIGN #6 TO Name$
4800 PRINTER IS 0
4805 PRINT "FILE-OF-FILES NAME: ";Name$
4810 PRINT
4815 PRINTER IS 16
4820 CALL Putline(8,0,"ENTER FILE-NAMES ONE AT A TIME & END WITH **")
4825 CALL Putline(10,0," ")
4830 LINPUT File$
4835 IF File$="**" THEN End_build
4840  IF POS(File$,":")=0 THEN File$=File$&":F9"
4845 PRINTER IS 0
4850 PRINT File$
4855 PRINTER IS 16
4860 PRINT #6;File$
4865 GOTO 4825
4870 End_build: !
4875 PRINT #6;"**"
4880 ASSIGN * TO #6
4885 GOTO Bye
4890 !
4895 Execute: !
4900 ASSIGN #6 TO Name$
4905 READ #6;File$
4910 IF File$="**" THEN End_execute
4915 ASSIGN #7 TO File$
4920 READ #7;Number_commands
4925  FOR J=1 TO Number_commands
4930  READ #7;Command$(J),Word$(J),Prefix$(J)
4935  NEXT J
4940 ASSIGN * TO #7
4945 IF Tf=0 THEN 4965
```

Subroutine "Mul_file" continued

```
4950 PRINTER IS 0
4955 PRINT Files
4960 PRINTER IS 16
4965 CALL Send
4970 GOTO 4905
4975 End_execute: !
4980 ASSIGN * TO #6
4985  GOTO Bye
4990 Listc:  PRINTER IS 0
4995  PRINT "CONTENTS OF FILE: ";Name$
5000  PRINT
5005  ASSIGN #6 TO Name$
5010  READ #6;S$
5015  IF S$="**" THEN GOTO Ext
5020  PRINT S$
5025  GOTO 5010
5030 Ext:  PRINTER IS 16
5035 GOTO Bye
5040 Trouble:  !
5045  IF ERRN=54 THEN Dup_file
5050  IF ERRN=56 THEN No_file
5055  STOP
5060 Dup_file:  !
5065  BEEP
5070  PRINT "FILE ALREADY EXISTS -- TRY AGAIN"
5075  WAIT 2000
5080  CALL Clear_line(7,10)
5085  GOTO 4710
5090 No_file:  !
5095  BEEP
5100  PRINT "BAD FILENAMES:  ";Name$;",  ";File$,"HALT!"
5105  STOP
5110 Bye: !
5115 SUBEND
```

Subroutine "Com _ group"

```
5140  SUB Com_group
5145  OPTION BASE 1
5150 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
5151 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
5152 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
5153 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
5154 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax,
Paxis
5155 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,D
8,D9
5156 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
5157 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
5158 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$[
20]
5159 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
5180  Menu=12
5185  CALL Clear
5190  IF Editf$<>" " THEN CALL Putline(0,0,CHR$(131)&"EDIT MODE"&CHR$(128))
5195  CALL Putline(0,23,"COMPUTER GROUP MENU")
5200  CALL Putline(2,3,"1 = PLOT LABEL")
5205  CALL Putline(2,30,"2 = FILE LABEL")
5210  CALL Putline(2,57,"3 = * UNUSED *")
5215  CALL Putline(3,3,"4 = * UNUSED *")
5220  CALL Putline(3,30,"5 = * UNUSED *")
5225  CALL Putline(3,57,"6 = LINEAR ARRAY PLOT")
5230  CALL Putline(4,3,"7 = LINEAR STEP PLOT")
5235  CALL Putline(4,30,"8 = LINEAR SPECIAL")
5240  CALL Putline(4,57,"9 = T-IM vs RE")
5245  CALL Putline(5,3,"10= T-RE vs IM*HZ")
5250  CALL Putline(5,30,"11= T-RE vs IM/HZ")
5255  CALL Putline(5,57,"12= T-COHERENCE")
5260  CALL Putline(6,3,"13= T-S/N")
5265  CALL Putline(6,30,"14= T-PHASE")
5270  CALL Putline(6,57,"15= T-LG MAG vs LG HZ")
5275  CALL Putline(7,3,"16= VOLTAGE HISTOGRAM")
5280 Inp: !
5285  CALL Putline(9,5,"INPUT YOUR CHOICE *")
5290  INPUT Com_choice
5295  IF (Com_choice=1) OR (Com_choice=2) THEN Labelp
5300  IF (Com_choice>=6) AND (Com_choice<=16) THEN Plots
5305  IF Com_choice=0 THEN Bye
5310  BEEP
5315  GOTO 5180
5320 Labelp: !
5325  ! L=Number_commands
5330  CALL Putline(11,10,"INPUT PAPER SIZE IN X,Y FORMAT (i.e. 8.5,11) :")
5335  INPUT Px,Py
5340  CALL Putline(13,10,"INPUT LABEL COORDINATES IN X,Y FORMAT:")
5345  INPUT Lx,Ly
5350  CALL Putline(15,10,"INPUT CHARACTER HEIGHT IN 64ths:")
5355  INPUT Chrsiz
5360  CALL Putline(17,10,"INPUT PEN # :")
5365  INPUT Pen
5370  Number_commands=Number_commands+1
5375  L=Number_commands
5380  IF Editf$="ADD" THEN L=Edit_line
5385  IF Editf$="ADD" THEN CALL Shifty(3)
5390  IF Com_choice=1 THEN 5405
5395  L$=File$
5400  GOTO 5415
5405  CALL Putline(19,10,"INPUT LABEL:")
5410  INPUT L$
5415  Command$(L)="*SYS*"
5420  IF Com_choice=1 THEN Prefix$(L)="1"
5425  IF Com_choice=2 THEN Prefix$(L)="2"
5430  Word$(L)="PLOTTER LABEL"
```

Subroutine "Com_group" continued

```
5435  L=L+1
5440  Command$(L)=VAL$(Px)&","&VAL$(Py)
5445  Prefix$(L)=VAL$(Lx)&","&VAL$(Ly)
5450  L=L+1
5455  Prefix$(L)=VAL$(Chrsiz)&","&VAL$(Pen)
5460  Command$(L)=L$
5465  CALL Clear_line(7,13)
5470  Word$(L)="ENDSYS 1"
5475  Number_commands=Number_commands+2
5480  GOTO Inp
5485 Plots:  !
5490  IF Com_choice=15 THEN Com_choice=21
5495  IF Com_choice=16 THEN Com_choice=37
5500  CALL Putline(11,10,"INPUT X & Y LABELS & TITLE SEPARATED BY COMMAS:")
5505  INPUT X$,Y$,T$
5510  CALL Putline(15,10,"INPUT PEN NUMBER:")
5515  INPUT Pn
5520  CALL Putline(17,10,"PLOT CODE 1=CRT  2=PRINTER  3=PLOTTER ?")
5525  INPUT Pc
5530  Number_commands=Number_commands+1
5535  L=Number_commands
5540  IF Editf$="ADD" THEN L=Edit_line
5545  IF Editf$="ADD" THEN CALL Shifty(3)
5550  Command$(L)="*SYS*"
5555  Prefix$(L)=VAL$(Com_choice)
5560  IF Com_choice>=9 THEN Prefix$(L)=VAL$(Com_choice+3)
5565  IF (Com_choice>=3) AND (Com_choice<=5) THEN Word$(L)="SEMILOG PLOT"
5570  IF (Com_choice>=6) AND (Com_choice<=8) THEN Word$(L)="LINEAR PLOT"
5575  IF (Com_choice>=9) AND (Com_choice<=15) THEN Word$(L)="TIED-DATA"
5580  IF Com_choice=16 THEN Word$(L)="HISTOGRAM"
5585  L=L+1
5590  Command$(L)=X$
5595  Prefix$(L)=Y$
5600  Word$(L)=T$
5605  L=L+1
5610  C=Com_choice-2
5615  IF Com_choice>=9 THEN C=Com_choice+3
5620  Command$(L)=VAL$(C)
5625  Prefix$(L)=VAL$(Pn)&","&VAL$(Pc)
5630  Word$(L)="ENDSYS PLOTS"
5635  Number_commands=Number_commands+2
5640  CALL Clear_line(7,13)
5645  GOTO Inp
5650 Bye: !
5655  SUBEND
```

Subroutine "Plot __ group"

```
10370 SUB Plot_group
10375 OPTION BASE 1
10380 COM Command$(*),Number_commands,Mode$,Menu,A5420,Word$(*),Prefix$(*)
10381 COM Editf$,Edit_line,B5420,SHORT Pdata(*),Nprint,Sdata(*),Nsave,Stat
10382 COM File$,SHORT State(*),Ry(*),Ix(*),Sz(*),Xline,Tc,Tf,Ti,Np,Slope,P1,P2
10383 COM SHORT Xstart,Xstep,Ryc(*),Ixc(*),Frc(*),Cp,Intercept,Frequency
10384 COM SHORT Plim,Pxo,Pxm,Pyo,Pym,Ploc,Xo,Xm,Yo,Ym,Pscale,Xmin,Xmax,Ymin,Ymax
,Paxis
10385 COM SHORT Cur,Poly,Lsf,Sflg,Region,Pflg,Pstart,Pstop,D1,D2,D3,D4,D5,D6,D7,
D8,D9
10386 COM SHORT Find_points,Paper,Blue,Black,Onn,Off,Skip,Device,Pager,Nopager
10387 COM SHORT Tpscale,Txmin,Txmax,Tymin,Tymax,Tnpts,Tx(*),Ty(*),Tflg,X60,Every
10388 COM SHORT Dpscale,Dxmin,Dxmax,Dymin,Dymax,Dnpts,Dflg,Plots,Cut,Nocut,Time$
[20]
10389 ! COM SHORT CPX,XTP(*),CPC,CTP(*)
10410 CALL Clear
10415 IF Editf$<>" " THEN CALL Putline(0,0,CHR$(131)&"EDIT MODE"&CHR$(128))
10420 CALL Putline(0,26,"PLOT GROUP MENU")
10425 CALL Putline(2,5,"1 = PLOT LIMITS")
10430 CALL Putline(2,25,"2 = PLOT LOCATION")
10435 CALL Putline(3,5,"3 = PLOT SCALE")
10440 CALL Putline(3,25,"4 = AXIS FLAG")
10445 CALL Putline(6,5,"INPUT YOUR CHOICE*")
10450 INPUT Pchoice
10455 IF Pchoice=0 THEN Bye
10460 Pchoice=Pchoice+25
10465 Number_commands=Number_commands+1
10470 IF Pchoice=29 THEN Px
10475 CALL Putline(8,5,"INPUT XO AND XM")
10480 LINPUT Pair1$
10485 CALL Putline(11,5,"INPUT YO AND YM")
10490 LINPUT Pair2$
10495 GOTO Kont
10500 Px: CALL Putline(8,5,"AXIS FLAG (0=NEW   1=NEW FOR OVERLAY  2=OVERLAY  3=L
AST OVERLAY)")
10505 INPUT X1
10510 X2=0
10515 Pair1$=VAL$(X1)&","&VAL$(X2)
10520 Pair2$="0,0"
10525 GOTO Kont
10530 Kont: !
10535 L=Number_commands
10540 IF (Editf$="ADD") OR (Editf$="REP") THEN L=Edit_line
10545 IF Editf$="ADD" THEN CALL Shifty(2)
10550 !
10555 Command$(L)="*SYS*"
10560 Prefix$(L)=VAL$(Pchoice)
10565 IF Pchoice=26 THEN Word$(L)="PLOT LIMITS"
10570 IF Pchoice=27 THEN Word$(L)="PLOT LOCATION"
10575 IF Pchoice=28 THEN Word$(L)="PLOT SCALE"
10580 IF Pchoice=29 THEN Word$(L)="AXIS FLAG"
10585 L=L+1
10590 Command$(L)=Pair1$
10595 Prefix$(L)=Pair2$
10600 Word$(L)="ENDSYS PLOT"
10605 CALL Clear_line(6,12)
10610 IF Editf$=" " THEN Number_commands=L
10615 IF Editf$="ADD" THEN Number_commands=Number_commands+1
10620 IF Editf$="REP" THEN Number_commands=Number_commands+0
10625 GOTO 10445
10630 Bye: !
10635 Menu=14
10640 SUBEND
```

Command Files

The HP9845 was used both to control the HP5420 for data acquisition and then to manipulate and graphically portray data sets. A collection of command files was written to control the computer in the execution of these tasks. The command files evolved from consolidations of small command sequences into longer files which execute an entire data collection program or perform an entire plot series. Although numerous small command files were created to perform specialized infrequent tasks, those described here constitute the major command files needed by the operator of the AC corrosion system.

Command files are created by calling appropriate sections of the unit group menu offered by system software. Once one has called the section on HP5420 set-up, for example, the user is queried as to which setup functions are desired. Upon answering these queries, a command file line is created. Once a sequence of commands has been created and checked, it may be named and saved for future use.

Data Acquisition Command Files

COMCEL - COMprehensive DEL1 Data Collection. Command file COMCEL controls the HP5420 in the acquisition of tied impedance and S/N data sets. The tied data sets consist of 1152 data points collected over five ranges of frequency: 0-25kHz, 0-3.2kHz, 0-256Hz and 0-3125Hz. The initial commands consist of setting all data points equal to zero and then setting up the HP5420 for data collection in the 0-25kHz range.

Impedance data is collected by determining the transfer function from an input voltage perturbation and an output current response. The transfer function, which is actually an admittance measurement, is

then inverted in the frequency domain to yield the impedance. Results of the 0-25kHz measurement are then stored on the floppy disk as file 6X, and the coherence function for this frequency range is determined. As shown in Appendix A, the coherence function may be manipulated to yield the S/N for that range of frequency. The S/N data for the 0-25kHz is stored as disk file 6C.

Subsequent set-up and data collection is accomplished in the same order for the remaining frequency ranges. Impedance data is stored in disk files 5X, 4X, 3X and 2X, respectively, while S/N data is stored in disk files 5C, 4C, 3C, and 2C, respectively. Upon completion and storage of impedance and S/N data for the lowest frequency range, the respective data sets are combined, throwing out low resolution overlapping data points. The resultant 1152 tied impedance data set is stored on the floppy disk as NAMETT while the tied S/N data set is stored as NAMESN. The entire command sequence takes roughly 12 minutes to execute.

SHTCEL - SHorT CELl Data Collection. Command file SHTCEL generates a tied impedance data set in the same manner as COMCEL. By eliminating the commands which generate S/N data, this sequence can be completed in about nine minutes.

Other Data Gathering Command Files. Because of the versatility of the main program, it is possible to create command sequences to perform any of the numerous operations within the capability of the HP5420. A number of others were created for the convenience of the author. This series, nicknamed FAST, were designed to provide a five decade frequency analysis as rapidly as possible. For example,

FASZTR delivers the impedance transfer function over a five-decade region, quickly. FASAUT delivers the auto power spectrum quickly, etc. A list of COMCEL, SHTCEL, FASZTR, AND FASAUT follows.

Command file "COMCEL"

```
// ////////////////////////
// COMMAND FILE COMCEL //
// ////////////////////////
// SET UP 5420 FOR 25.6KHZ BANDWIDTH
   LL;                                      LOC LOCKOUT
   *SYS*                  20                ZERO DATA SETS
   0 RS;                  0                 RESET
   5 FR;                  5                 FREQ
   20,1 AV;               20,1              AVG
   2 SG;                  2                 SGHL
   1 TG;                  1                 TRIG
   0 CH;                  0                 CHAN #
   1KV   RG;              1VOLT             RANGE
   1 AC;                  1                 AC,DC
   1 CA;                  1                 CAL
   0HD CF;                0HZ               CENT FREQ
   25.6KV  BW;            25.6KHZ           BW
   MR;                                      MAX RATE
   1 CH;                  1                 CHAN #
   5 FR;                  5                 FREQ
// START RUN FOR 25.6KHZ BANDWIDTH
   ST;                                      START
   IR;                                      IM/RE
// CONVERT ADMITTANCE TO IMPEDANCE
   -1 DV;                 -1                /
// SAVE IMPEDANCE FUNCTION
   501 SA;                501               SAVE
   *SYS*                  9                 6X
   -1 DV;                 -1                /
   2 FM;                  2                 FORMAT
   0 TC;                  0                 TRACE
// DETERMINE S/N
   CO;                                      COHER
   MG;                                      MAG
   2 TC;                  2                 TRACE
   ,1,0 SB;               ,1,0              -
   DV;                                      /
     LM;                                    LOG MAG
   1 FM;                  1                 FORMAT
// SAVE S/N DATA FOR 25.6KHZ BAND WIDTH
   501 SA;                501               SAVE
   *SYS*                  9                 6C
// SET UP AND COLLECT DATA FOR 3.2KHZ BAND WIDTH0
   3.2KV  BW;             3.2KHZ            BW
   1 CH;                  1                 CHAN #
     TN;                                    TRANS
   ST;                                      START
   IR;                                      IM/RE
   -1 DV;                 -1                /
   501 SA;                501               SAVE
   *SYS*                  9                 5X
   -1 DV;                 -1                /
   2 FM;                  2                 FORMAT
   0 TC;                  0                 TRACE
   CO;                                      COHER
   MG;                                      MAG
   2 TC;                  2                 TRACE
   ,1,0 SB;               ,1,0              -
   DV;                                      /
     LM;                                    LOG MAG
   1 FM;                  1                 FORMAT
   501 SA;                501               SAVE
   *SYS*                  9                 5C
// SET UP AND COLLECT DATA FOR 256 HZ
```

Command File "COMCEL" continued

```
256HD BW;                 256HZ                    BW
1 CH;                     1                        CHAN #
  TN;                                              TRANS
ST;                                                START
IR;                                                IM/RE
-1 DV;                    -1                       /
501 SA;                   501                      SAVE
*SYS*                     9                        4X
-1 DV;                    -1                       /
2 FM;                     2                        FORMAT
0 TC;                     0                        TRACE
CO;                                                COHER
MG;                                                MAG
2 TC;                     2                        TRACE
,1,0 SB;                  ,1,0                     -
DV;                                                /
  LM;                                              LOG MAG
1 FM;                     1                        FORMAT
501 SA;                   501                      SAVE
*SYS*                     9                        4C
// SET UP AND COLLECT DATA FOR 25 HZ
25HD BW;                  25HZ                     BW
1 CH;                     1                        CHAN #
  TN;                                              TRANS
ST;                                                START
IR;                                                IM/RE
-1 DV;                    -1                       /
501 SA;                   501                      SAVE
*SYS*                     9                        3X
-1 DV;                    -1                       /
2 FM;                     2                        FORMAT
0 TC;                     0                        TRACE
CO;                                                COHER
MG;                                                MAG
2 TC;                     2                        TRACE
,1,0 SB;                  ,1,0                     -
DV;                                                /
  LM;                                              LOG MAG
1 FM;                     1                        FORMAT
501 SA;                   501                      SAVE
*SYS*                     9                        3C
// SET UP AND COLLECT DATA FOR 3.125HZ
3.125HD BW;               3.125HZ                  BW
1 CH;                     1                        CHAN #
  TN;                                              TRANS
ST;                                                START
IR;                                                IM/RE
-1 DV;                    -1                       /
501 SA;                   501                      SAVE
*SYS*                     9                        2X
-1 DV;                    -1                       /
2 FM;                     2                        FORMAT
0 TC;                     0                        TRACE
CO;                                                COHER
MG;                                                MAG
2 TC;                     2                        TRACE
,1,0 SB;                  ,1,0                     -
DV;                                                /
  LM;                                              LOG MAG
1 FM;                     1                        FORMAT
501 SA;                   501                      SAVE
*SYS*                     9                        2C
// TIE IMPEDANCE DATA SETS
*SYS*                     11                       X
*SYS*                     10                       NAMETT
// TIE S/H DATA SETS
*SYS*                     11                       C
```

Command file "COMCEL" concluded

132	*SYS*	18	NAMESN
133	GL;		GOTO LOCAL

Command file "SHTCEL"

```
// ///////////////////////
// COMMAND FILE SHTCEL //
// ///////////////////////
// SET UP 5420 FOR 25.6KHZ BANDWIDTH
   LL;                                             LOC LOCKOUT
   *SYS*                    20                     ZERO DATA SETS
   0 RS;                    0                      RESET
   5 FR;                    5                      FREQ
   20,1 AV;                 20,1                   AVG
   2 SG;                    2                      SGNL
   1 TG;                    1                      TRIG
   0 CH;                    0                      CHAN #
   .1KV   RG;               .1VOLT                 RANGE
   1 AC;                    1                      AC,DC
   1 CA;                    1                      CAL
   0HD CF;                  0HZ                    CENT FREQ
   25.6KV  BW;              25.6KHZ                BW
   MR;                                             MAX RATE
   1 CH;                    1                      CHAN #
   5 FR;                    5                      FREQ
// START RUN FOR 25.6KHZ BANDWIDTH
   ST;                                             START
   IR;                                             IM/RE
// CONVERT ADMITTANCE TO IMPEDANCE
   -1 DV;                   -1                     /
// SAVE IMPEDANCE FUNCTION
   501 SA;                  501                    SAVE
   *SYS*                    9                      6X
// SET UP AND COLLECT DATA FOR 3.2KHZ BAND WIDTH0
   3.2KV  BW;               3.2KHZ                 BW
   1 CH;                    1                      CHAN #
     TN;                                           TRANS
   ST;                                             START
   IR;                                             IM/RE
   -1 DV;                   -1                     /
   501 SA;                  501                    SAVE
   *SYS*                    9                      5X
// SET UP AND COLLECT DATA FOR 256 HZ
   256HD BW;                256HZ                  BW
   1 CH;                    1                      CHAN #
     TN;                                           TRANS
   ST;                                             START
   IR;                                             IM/RE
   -1 DV;                   -1                     /
   501 SA;                  501                    SAVE
   *SYS*                    9                      4X
// SET UP AND COLLECT DATA FOR 25 HZ
   25HD BW;                 25HZ                   BW
   1 CH;                    1                      CHAN #
     TN;                                           TRANS
   ST;                                             START
   IR;                                             IM/RE
   -1 DV;                   -1                     /
   501 SA;                  501                    SAVE
   *SYS*                    9                      3X
// SET UP AND COLLECT DATA FOR 3.125HZ
   3.125HD BW;              3.125HZ                BW
   1 CH;                    1                      CHAN #
     TN;                                           TRANS
   ST;                                             START
   IR;                                             IM/RE
   -1 DV;                   -1                     /
   501 SA;                  501                    SAVE
   *SYS*                    9                      2X
```

Command File "SHTCEL" concluded

.
.

```
// TIE IMPEDANCE DATA SETS
   *SYS*                  11                     X
   *SYS*                  18                     NAME 'T
     GL;                                         GOTC LOCAL
```

Command File "FASZTR"

LL;		LOC LOCKOUT
SYS	20	ZERO DATA SETS
0 RS;	0	RESET
5 FR;	5	FREQ
20,1 AV;	20,1	AVG
2 SG;	2	SGNL
1 TG;	1	TRIG
0 CH;	0	CHAN #
.1KV RG;	.1VOLT	RANGE
0 AC;	0	AC,DC
0HD CF;	0HZ	CENT FREQ
25.6KV BW;	25.6KHZ	BW
MR;		MAX RATE
ST;		START
1 TC;	1	TRACE
1 FM;	1	FORMAT
IR;		IM/RE
-1 DV;	-1	/
501 SA;	501	SAVE
SYS	9	6X
3.2KV BW;	3.2KHZ	BW
ST;		START
-1 DV;	-1	/
501 SA;	501	SAVE
SYS	9	5X
256HD BW;	256HZ	BW
ST;		START
-1 DV;	-1	/
501 SA;	501	SAVE
SYS	9	4X
25HD BW;	25HZ	BW
ST;		START
-1 DV;	-1	/
501 SA;	501	SAVE
SYS	9	3X
3.125HD BW;	3.125HZ	BW
ST;		START
-1 DV;	-1	/
501 SA;	501	SAVE
SYS	9	2X
SYS	11	X
SYS	18	NAMETT
GL;		GOTO LOCAL

Command File "FASAUT"

LL;		LOC LOCKOUT
SYS	20	ZERO DATA SETS
0 RS;	0	RESET
2 FR;	2	FREQ
20,1 AV;	20,1	AVG
2 SG;	2	SGNL
1 TG;	1	TRIG
0 CH;	0	CHAN #
.1KV RG;	.1VOLT	RANGE
0 AC;	0	AC,DC
0HD CF;	0HZ	CENT FREQ
25.6KV BW;	25.6KHZ	BW
MR;		MAX RATE
ST;		START
1 TC;	1	TRACE
1 FM;	1	FORMAT
IR;		IM/RE
,100 MY;	,100	*
501 SA;	501	SAVE
SYS	9	6X
3.2KV BW;	3.2KHZ	BW
ST;		START
,12.5 MY;	,12.5	*
501 SA;	501	SAVE
SYS	9	5X
256HD BW;	256HZ	BW
ST;		START
501 SA;	501	SAVE
SYS	9	4X
25HD BW;	25HZ	BW
ST;		START
,.097656 MY;	,.097656	*
501 SA;	501	SAVE
SYS	9	3X
3.125HD BW;	3.125HZ	BW
ST;		START
,.012207 MY;	,.012207	*
501 SA;	501	SAVE
SYS	9	2X
SYS	11	X
SYS	18	NAMESG
GL;		GOTO LOCAL

Plotting Command Files

PLDZTR - Plot Difference between Impedance Transfer function: the purpose of PLDZTR is to provide a comprehensive graphical comparison between a theoretical data set and actual data. The plot is auto-scaled to contain the theoretical data and the experimental is plotted as an overlay. All five plot formats, Imaginary vs. Real, Real vs. Imaginary x Frequency, Real vs. Imaginary/Frequency, magnitude vs. frequency, and phase angle vs. frequency are plotted. A least squares fit is made of the straight line plots and an equation for the lines is given. An analysis is then performed on the graphical presentations to show what each of them pick the values of a three-element network fitting the experimental data.

Other Plotting Command Files. Other plotting command sequences were created for convenience. The PLAxxx series imply the plot is auto-scaled. The PLOxxx series imply that 0,0 is the lower left-hand origin. The PLSxxx series imply that a peculiar scaling was used, etc. The last three digits indicate the type of plot, e.g. ZTR, impedance transfer function; IVR, imaginary vs. real; BOD, Bode plots, etc. Listings of PLDZTR, PLAIVR, PL01VR, PLABOD and PLS BOD follow.

Command File "PLDZTR"

```
// //////////////////// //
// COMMAND FILE PLDZTR
// ///////////////////// //
// AUTO SCALE THIS PLOT SERIES
    *SYS*                  26                         PLOT LIMITS
    0,0                    0,0                        ENDSYS PLOT
    *SYS*                  27                         PLOT LOCATION
    0,0                    0,0                        ENDSYS PLOT
    *SYS*                  28                         PLOT SCALE
    0,0                    0,0                        ENDSYS PLOT
    *SYS*                  29                         AXIS FLAG
    1,0                    0,0                        ENDSYS PLOT
// RECALL THEORETICAL DATA
    *SYS*                  19                         THEODA
// PLOT THEORETICAL DATA
    *SYS*                  12                         TIED-DATA
    REAL (OHMS)            IMAGINARY (OHMS)           CELL IMPEDANCE
    12                     4,2                        ENDSYS PLOTS
// RECALL EXPERIMENTAL DATA
    *SYS*                  19                         NAMETT
    *SYS*                  45
// PLOT EXPERIMENTAL DATA
    *SYS*                  29                         AXIS FLAG
    3,0                    0,0                        ENDSYS PLOT
    *SYS*                  12                         TIED-DATA
    REAL (OHMS)            IMAGINARY (ONMS)           CELL IMPEDANCE
    12                     4,2                        ENDSYS 3,4,5,6,7,8
// AUTO SCALE
    *SYS*                  28                         PLOT SCALE
    0,0                    0,0                        ENDSYS PLOT
    *SYS*                  29                         AXIS FLAG
    1,0                    0,0                        ENDSYS PLOT
// RECALL THEORETICAL DATA
    *SYS*                  19                         THEODA
// PLOT THEORETICAL DATA
    *SYS*                  13                         TIED-DATA
    -IMAG*HZ               REAL                       CELL IMPEDANCE
    13                     4,2                        ENDSYS PLOTS
// RECALL EXPERIMENTAL DATA
    *SYS*                  19                         NAMETT
    *SYS*                  45
// PLOT EXPERIMENTAL DATA
    *SYS*                  29                         AXIS FLAG
    3,0                    0,0                        ENDSYS PLOT
    *SYS*                  13                         TIED-DATA
    -IMAG*HZ               REAL                       CELL IMPEDANCE
    13                     4,2                        ENDSYS 3,4,5,6,7,8
// LEAST SQUARES FIT
    *SYS*                  30                         LEAST SQ.
    *SYS*                  28                         PLOT SCALE
    0,0                    0,0                        ENDSYS PLOT
    *SYS*                  29                         AXIS FLAG
    0,0                    0,0                        ENDSYS PLOT
    *SYS*                  13                         TIED-DATA
    -IMAG*HZ               REAL                       CELL IMPEDANCE
    13                     4,2                        ENDSYS PLOTS
// AUTO SCALE
    *SYS*                  28                         PLOT SCALE
    0,0                    0,0                        ENDSYS PLOT
    *SYS*                  29                         AXIS FLAG
    1,0                    0,0                        ENDSYS PLOT
// RECALL THEORETICAL DATA
    *SYS*                  19                         THEODA
// PLOT THEORETICAL DATA
```

Command File "PLDZTR" continued

```
    *SYS*                  14                     TIED-DATA
    -IMAG/HZ               REAL                   CELL IMPEDANCE
    14                     4,2                    ENDSYS PLOTS
 // RECALL EXPERIMENTAL DATA
    *SYS*                  19                     NAMETT
    *SYS*                  45
 // PLOT EXPERIMENTAL DATA
    *SYS*                  29                     AXIS FLAG
    3,0                    0,0                    ENDSYS PLOT
    *SYS*                  14                     TIED-DATA
    -IMAG/HZ               REAL                   CELL IMPEDANCE
    14                     4,2                    ENDSYS 3,4,5,6,7,8
 // LEAST SQUARES FIT
    *SYS*                  30                     LEAST SQ.
    *SYS*                  28                     PLOT SCALE
    0,0                    0,0                    ENDSYS PLOT
    *SYS*                  29                     AXIS FLAG
    0,0                    0,0                    ENDSYS PLOT
    *SYS*                  14                     TIED-DATA
    -IMAG/HZ               REAL                   CELL IMPEDANCE
    14                     4,2                    ENDSYS PLOTS
 // AUTO SCALE
    *SYS*                  28                     PLOT SCALE
    0,0                    0,0                    ENDSYS PLOT
    *SYS*                  29                     AXIS FLAG
    1,0                    0,0                    ENDSYS PLOT
 // RECALL THEORETICAL DATA
    *SYS*                  19                     THEODA
 // PLOT THEORETICAL DATA
    *SYS*                  24
    LOG FREQUENCY          LOG MAGNITUDE (DB)     CELL IMPEDANCE
    24                     4,2                    ENDSYS PLOTS
 // RECALL EXPERIMENTAL DATA
    *SYS*                  19                     NAMETT
    *SYS*                  45
 // PLOT EXPERIMENTAL DATA
    *SYS*                  29                     AXIS FLAG
    3,0                    0,0                    ENDSYS PLOT
    *SYS*                  24
    LOG FREQUENCY          LOG MAGNITUDE (DB)     CELL IMPEDANCE
    24                     4,2                    ENDSYS 3,4,5,6,7,8
 // AUTO SCALE
    *SYS*                  28                     PLOT SCALE
    0,0                    0,0                    ENDSYS PLOT
    *SYS*                  29                     AXIS FLAG
    1,0                    0,0                    ENDSYS PLOT
 // RECALL THEORETICAL DATA
    *SYS*                  19                     THEODA
 // PLOT THEORETICAL DATA
    *SYS*                  17                     TIED-DATA
    LOG FREQUENCY          PHASE ANGLE (DEG)      PHASE RESPONSE
    17                     4,2                    ENDSYS PLOTS
 // RECALL EXPERIMENTAL DATA
    *SYS*                  19                     NAMETT
    *SYS*                  45
 // PLOT EXPERIMENTAL DATA
    *SYS*                  29                     AXIS FLAG
    3,0                    0,0                    ENDSYS PLOT
    *SYS*                  17                     TIED-DATA
    .                      .                      .
    17                     4,2                    ENDSYS PLOTS
    *SYS*                  43                     ANALYSIS
```

Command File "PLAIVR"

```
// //////////////////// //
// COMMAND FILE PLAIVR //
// //////////////////// //
// AUTO SCALE THIS PLOT SERIES
   *SYS*                26                       PLOT LIMITS
   0,0                  0,0                      ENDSYS PLOT
   *SYS*                27                       PLOT LOCATION
   0,0                  0,0                      ENDSYS PLOT
   *SYS*                28                       PLOT SCALE
   0,0                  0,0                      ENDSYS PLOT
   *SYS*                29                       AXIS FLAG
   0,0                  0,0                      ENDSYS PLOT
// RECALL EXPERIMENTAL DATA SET
   *SYS*                19                       HAMETT
// REMOVE 60 HZ HARMONICS
   *SYS*                44                       HARMONICS
// PLOT COMBED DATA SET IN THE REGION OF INTEREST
   *SYS*                29                       AXIS FLAG
   1,0                  0,0                      ENDSYS PLOT
   *SYS*                12                       TIED-DATA
   REAL (OHMS)          IMAGINARY (OHMS)         CELL IMPEDANCE
   12                   4,2                      ENDSYS 3,4,5,6,7,8
// RECALL THEORETICAL DATA
   *SYS*                19                       THEODA
// REMOVE 60 HZ HARMONICS
   *SYS*                44                       HARMONICS
// PLOT THEORETICAL DATA
   *SYS*                29                       AXIS FLAG
   3,0                  0,0                      ENDSYS PLOT
   *SYS*                12                       TIED-DATA
   .                    .                        .
   12                   4,2                      ENDSYS PLOTS
```

Command File "PL01VR"

```
// //////////////////// //
// COMMAND FILE PL0IVR //
// //////////////////// //
// QUADRANT SCALE THIS PLOT SERIES
   *SYS*                 26                          PLOT LIMITS
   0,0                   0,0                         ENDSYS PLOT
   *SYS*                 27                          PLOT LOCATION
   0,0                   0,0                         ENDSYS PLOT
   *SYS*                 28                          PLOT SCALE
   0,*                   *,0                         ENDSYS PLOT
   *SYS*                 29                          AXIS FLAG
   0,0                   0,0                         ENDSYS PLOT
// RECALL EXPERIMENTAL DATA SET
   *SYS*                 19                          HAMETT
// REMOVE 60 HZ HARMONICS
   *SYS*                 44                          HARMONICS
// PLOT COMBED DATA SET IN THE REGION OF INTEREST
   *SYS*                 29                          AXIS FLAG
   1,0                   0,0                         ENDSYS PLOT
   *SYS*                 12                          TIED-DATA
   REAL (OHMS)           IMAGINARY (OHMS)            CELL IMPEDANCE
   12                    4,2                         ENDSYS 3,4,5,6,7,8
// RECALL THEORETICAL DATA
   *SYS*                 19                          THEODA
// REMOVE 60 HZ HARMONICS
   *SYS*                 44                          HARMONICS
// PLOT THEORETICAL DATA
   *SYS*                 29                          AXIS FLAG
   3,0                   0,0                         ENDSYS PLOT
   *SYS*                 12                          TIED-DATA
   .                     .                           .
   12                    4,2                         ENDSYS PLOTS
```

Command File "PLABOD"

```
// //////////////////// //
// COMMAND FILE PLABOD //
// //////////////////// //
// AUTO SCALE THIS PLOT SERIES
   0,0                 0,0                    ENDSYS PLOT
   *SYS*               27                     PLOT LOCATION
   0,0                 0,0                    ENDSYS PLOT
   *SYS*               28                     PLOT SCALE
   0,0                 0,0                    ENDSYS PLOT
   *SYS*               29                     AXIS FLAG
   0,0                 0,0                    ENDSYS PLOT
// RECALL EXPERIMENTAL DATA SET
   *SYS*               19                     NAMETT
// REMOVE 60 HZ HARMONICS
   *SYS*               44                     HARMONICS
// PLOT MAGNITUDE
   *SYS*               29                     AXIS FLAG
   1,0                 0,0                    ENDSYS PLOT
   *SYS*               24
   LOG FREQUENCY       LOG MAGNITUDE          CELL IMPEDANCE
   24                  4,2                    ENDSYS PLOTS
// RECALL THEORETICAL DATA
   *SYS*               19                     THEODA
// REMOVE 60 HZ HARMONICS
   *SYS*               44                     HARMONICS
// PLOT THEORETICAL DATA
   *SYS*               29                     AXIS FLAG
   3,0                 0,0                    ENDSYS PLOT
   *SYS*               24
   .                   .                      .
   24                  4,2                    ENDSYS PLOTS
// AUTO SCALE
   *SYS*               28                     PLOT SCALE
   0,0                 0,0                    ENDSYS PLOT
   *SYS*               29                     AXIS FLAG
   0,0                 0,0                    ENDSYS PLOT
// RECALL EXPERIMENTAL DATA
   *SYS*               19                     NAMETT
// REMOVE 60 HZ HARMONICS
   *SYS*               44                     HARMONICS
// PLOT PHASE ANGLE
   *SYS*               29                     AXIS FLAG
   1,0                 0,0                    ENDSYS PLOT
   *SYS*               17                     TIED-DATA
   LOG FREQUENCY       PHASE ANGLE (DEG)      PHASE RESPONSE
   17                  4,2                    ENDSYS PLOTS
// RECALL THEORETICAL DATA
   *SYS*               19                     THEODA
// REMOVE 60 HZ HARMONICS
   *SYS*               44                     HARMONICS
// PLOT THEORETICAL DATA
   *SYS*               29                     AXIS FLAG
   3,0                 0,0                    ENDSYS PLOT
   *SYS*               17                     TIED-DATA
   .                   .                      .
   17                  4,2                    ENDSYS PLOTS
```

Command File "PLSBOD"

```
// /////////////////// //
// COMMAND FILE PLSBOD //
// /////////////////// //
// AUTO SCALE THIS PLOT SERIES
    0,0                       0,0                   ENDSYS PLOT
    *SYS*                     27                    PLOT LOCATION
    0,0                       0,0                   ENDSYS PLOT
    *SYS*                     20                    PLOT SCALE
    0,0                       0,0                   ENDSYS PLOT
    *SYS*                     29                    AXIS FLAG
    0,0                       0,0                   ENDSYS PLOT
// RECALL EXPERIMENTAL DATA SET
    *SYS*                     19                    NAMETT
// REMOVE 60 HZ HARMONICS
    *SYS*                     44                    HARMONICS
// PLOT MAGNITUDE
    *SYS*                     29                    AXIS FLAG
    1,0                       0,0                   ENDSYS PLOT
    *SYS*                     24
    LOG FREQUENCY             LOG MAGNITUDE         CELL IMPEDANCE
    24                        4,2                   ENDSYS PLOTS
// SMOOTH DATA AND PLOT AGAIN
    *SYS*                     37                    10
    *SYS*                     29                    AXIS FLAG
    3,0                       0,0                   ENDSYS PLOT
    *SYS*                     24
    .                         .                     .
    24                        4,2                   ENDSYS PLOTS
// RECALL EXPERIMENTAL DATA
    *SYS*                     19                    NAMETT
// REMOVE 60 HZ HARMONICS
    *SYS*                     44                    HARMONICS
// PLOT PHASE ANGLE
    *SYS*                     20                    PLOT SCALE
    0,0                       0,0                   ENDSYS PLOT
    *SYS*                     29                    AXIS FLAG
    1,0                       0,0                   ENDSYS PLOT
    *SYS*                     17                    TIED-DATA
    LOG FREQUENCY             PHASE ANGLE (DEG)     PHASE RESPONSE
    17                        4,2                   ENDSYS PLOTS
// SMOOTH DATA AND PLOT AGAIN
    *SYS*                     37                    10
    *SYS*                     29                    AXIS FLAG
    3,0                       0,0                   ENDSYS PLOT
    *SYS*                     17                    TIED-DATA
    .                         .                     .
    17                        4,2                   ENDSYS PLOTS
```

System Codes

The system codes were mentioned in Table D-1. There are a series of numbers used in the command files to indicate that a particular plotting format or type of analysis is to be generated. Most of the codes are inserted simply by answering the prompts from the main program. They may also be inserted into a command file using the edit mode (Key 9) and responding with "Q" when asked whether one wishes to insert or replace a line. The computer will respond with three input prompts. The respective inputs should be *SYS*, system code number and any word which explains the purpose of the code. (The last input is for comment on the command file listing.) Table D-3 lists the functions of the various system codes.

TABLE D-3 SYSTEM CODES

PREFIX	CODE	FUNCTION
SYS	1	LABEL ON PLOTTER
SYS	2	FILENAME ON PLOTTER
SYS	3	SEMILOG ARRAY PLOT
SYS	4	SEMILOG START/STEP PLOT
SYS	5	SEMILOG IMAG*HZ PLOT
SYS	6	LINEAR ARRAY PLOT
SYS	7	LINEAR START/STEP PLOT
SYS	8	LINEAR IMAG*HZ PLOT
SYS	9	SAVE SHORT DATA SET
SYS	10	RECALL SHORT DATA SET
SYS	11	TIE DATA FILES
SYS	12	TIED Im vs Re PLOT
SYS	13	TIED Re vs Im*Hz PLOT
SYS	14	TIED Re vs Im/Hz PLOT
SYS	15	TIED COHERENCE PLOT
SYS	16	TIED S/N PLOT
SYS	17	TIED PHASE PLOT
SYS	18	SAVE TIED DATA SET
SYS	19	RECALL TIED DATA SET
SYS	20	CREATE ZEROED SHORT DATA SETS
SYS	21	RENAME SHORT DATA SETS
SYS	22	SWAP REAL AND IMAGINARY ARRAYS
SYS	23	SCALE REAL AND IMAGINARY ARRAYS
SYS	24	BODE PLOT

Table D-3--continued.

PREFIX	CODE	FUNCTION
SYS	25	SET PLOT SCALE LIMITS FROM LAST PLOT
SYS	26	SET PLOT LIMITS FOR UNIT
SYS	27	SET PLOT LOCATION ON UNIT
SYS	28	SET PLOT SCALE FACTOR
SYS	29	SET AXIS FLAG
SYS	30	ENABLE LEAST SQUARES FIT ON DATA
SYS	31	ENABLE POLYNOMIAL FIT ON DATA
SYS	32	SMOOTH REAL SHORT DATA
SYS	33	SMOOTH IMAGINARY SHORT DATA
SYS	34	SMOOTH BOTH SHORT
SYS	35	SMOOTH REAL TIED
SYS	36	SMOOTH IMAGINARY TIED
SYS	37	SMOOTH BOTH TIED
SYS	38	MOVE SHORT DATA TO TIED DATA
SYS	39	DELETE FILES
SYS	40	VOLTAGE HISTOGRAM
SYS	41	ENABLE CURSOR FOR EXAMINATION OF DATA
SYS	42	SET REGION OF INTREST ON NEXT PLOT
SYS	43	ENABLE ANALYSIS
SYS	44	REMOVE 60 HZ. HARMONICS
SYS	45	ENABLE LOCAL ANALYSIS
SYS	46	SHORTEN A LONG TIED DATA SET

GRAFIT

Analysis capability for three-element networks is built into the main program for the HP9845. However, when the author performed the analysis on the 430 stainless steel experiments, he realized that a five-element network model would be required. Since he no longer had access to an HP9845, a BASIC program was written compatible with an HP85 microcomputer and HP7225 graphics plotter. This program, entitled GRAFIT, is actually a generic plotting program which creates camera-ready plots in linear or semi-logarithmic format which accepts data stored on tape, entered as x, y pairs, or generated by an appropriate function subroutine. In this case, the function subroutine generates the real and imaginary parts of the network impedance as a function of frequency. A listing of GRAFIT follows.

"GRAFIT" Program Listing

```
10 CLEAR
15 PLOTTER IS 705
20 OPTION BASE 1
25 DIM D$[1],B$[1],O$[1],L$[18],I(251),J(251),K(251)
30 D$="O"
31 R$="NEW"
35 D1=10
40 DISP "********************************"
45 DISP "*********PROGRAM GRAFIT*********"
50 DISP "********************************"
55 DISP
60 DISP "     GRAFIT CREATES CAMERA-READY"
65 DISP "GRAPHS ON THE HP7225 PLOTTER."
70 DISP "YOU MAY PRODUCE THE GRAPHICS IN"
75 DISP "EITHER AN INTERACTIVE MODE OR BY"
80 DISP "USING A DEFAULT MODE WHICH GIVES"
85 DISP "YOU A PARTICULAR FORMAT.  IF YOU"
90 DISP "HAVE NOT ALREADY DONE SO, TURN"
95 DISP "ON THE PLOTTER AND PLACE A "
100 DISP "PIECE OF PAPER ON THE PLOTTING"
105 DISP "SURFACE."
110 DISP "IF YOU WISH TO OPERATE IN THE"
115 DISP "DEFAULT MODE, PRESS 'D': IF YOU"
120 DISP "WISH TO INTERACT THEN PRESS ANY"
125 DISP "OTHER CHARACTER."
130 INPUT D$
135 IF D$="D" THEN GOTO 745
140 CLEAR
145 DISP "********************************"
150 DISP "*********GRAPHICS PLAN*********"
155 DISP "********************************"
160 DISP
165 DISP "PAPER DIMENSIONS IN INCHES"
170 DISP "     HORIZONTAL?"
175 INPUT H
180 DISP "     VERTICAL?"
185 INPUT V
190 CLEAR
195 DISP "MARGINS"
200 DISP "(INCHES FROM RESPECTIVE EDGE)"
205 DISP "     TOP?"
210 INPUT T1
215 DISP "     BOTTOM?"
220 INPUT B1
225 DISP "     RIGHT?"
230 INPUT R1
235 DISP "     LEFT?"
240 INPUT L1
245 CLEAR
250 DISP "GRID PLACEMENT"
255 DISP "(INCHES FROM RESPECTIVE MARGIN)"
260 DISP "     TOP?"
265 INPUT T2
270 DISP "     BOTTOM?"
275 INPUT B2
280 DISP "     RIGHT?"
285 INPUT R2
290 DISP "     LEFT?"
295 INPUT L2
```

"GRAFIT" Program Listing--continued.

```
300 CLEAR
305 DISP "FRAME THE GRID AREA? (Y OR N)"
310 INPUT G$
315 CLEAR
320 DISP "**************AXES*************'
325 DISP "SCALING"
330 DISP "  XAXIS"
335 DISP "      LOG SCALE? (Y OR N)"
340 INPUT Q1$
345 DISP "      XMIN?"
350 INPUT X1
355 DISP "      XMAX?"
360 INPUT X2
365 DISP "      YAXIS INTERCEPT"
370 INPUT I3
375 DISP "  YAXIS"
380 DISP "      LOG SCALE? (Y OR N)"
385 INPUT Q2$
390 DISP "      YMIN"
395 INPUT Y1
400 DISP "      YMAX"
405 INPUT Y2
410 DISP "      XAXIS INTERCEPT"
415 INPUT I4
420 CLEAR
425 DISP "TIC MARKS"
430 DISP "  XAXIS MAJOR DIVISIONS"
435 INPUT I1
440 DISP "      # MINOR TICS/MAJOR TIC?"
445 INPUT I5
450 DISP "  YAXIS MAJOR DIVISIONS"
455 INPUT I2
460 DISP "      # MINOR TICS/MAJOR TIC?"
465 INPUT I6
470 IF D1=3 THEN GOTO 830
475 CLEAR
480 ! NUMERIC AXIS LABELING
485 DISP "AXIS LABELING-NUMERIC"
490 DISP "  CHARACTER SIZE"
495 DISP "      CHARACTER HT- % OF GRID HT"
500 INPUT C8
505 DISP "      CHAR ASPECT RATIO (W/H)"
510 INPUT A1
515 DISP "      CHAR SLANT (DEG TO VERT)"
520 INPUT S8
525 DISP "  LABEL LOCATION"
530 DISP "      XLABELS ABOVE OR BELOW GRID? (A OR B)'
535 INPUT Q3$
540 DISP "      YLABELS LEFT OR RIGHT OF GRID? (L OR R)"
545 INPUT Q4$
550 CLEAR
555 ! ALPHANUMERIC AXIS LABELING
560 DISP "AXIS TITLES"
565 DISP "  CHARACTER SIZE"
570 DISP "      CHARACTER HT- % OF GRID HT"
575 INPUT C7
580 DISP "      CHAR ASPECT RATIO (W/H)"
585 INPUT A2
590 DISP "      CHAR SLANT (DEG TO VERT)"
595 INPUT S7
```

"GRAFIT" Program Listing--continued.

```
600 CLEAR
605 DISP "  XAXIS TITLE (<15 CHAR)"
610 INPUT X$
615 DISP "      TITLE ABOVE OR BELOW GRID?"
620 DISP "      (A OR B?)"
625 INPUT Q5$
630 CLEAR
635 DISP "  YAXIS TITLE (<15 CHAR)"
640 INPUT Y$
645 DISP "      TITLE LEFT OR RIGHT OF GRID"
650 DISP "      (L OR R?)"
655 INPUT Q6$
660 CLEAR
665 DISP "PLOT TITLE"
670 DISP "  CHARACTER SIZE"
675 DISP "      CHARACTER HT - % OF GRID HT"
680 INPUT C6
685 DISP "      CHAR ASPECT RATIO (W/H)"
690 INPUT A3
695 DISP "CHAR SLANT (DEG TO VERT)"
700 INPUT S6
705 DISP " TITLE VERBAGE?  (<15 CHAR)"
710 INPUT T$
715 DISP "  TITLE ABOVE OR BELOW GRID"
720 DISP "   (A OR B?)"
725 INPUT Q7$
730 GOSUB 1060
735 GOSUB 1575
740 GOSUB 1995
745 CLEAR
750 DISP "YOU HAVE THE FOLLOWING CHOICES"
755 DISP "WITH REGARD TO PLOT FORMAT:"
756 DISP "      0.  GRID PLOTTED; NEW DATA."
760 DISP "      1.  SAME PLOT FORMAT;"
765 DISP "      NEW DATA."
770 DISP "      2.  SAME PLOT FORMAT;"
775 DISP "      NEW SCALE, NEW DATA."
780 DISP "      3.  USER SPECIFIED FORMAT"
785 DISP
790 DISP "  WHICH DO YOU WANT?  (0,1,2,3)"
795 INPUT D1
796 D1=D1+1
800 IF D1=4 THEN GOTO 845
805 ASSIGN# 1 TO "PLTDAT:D700"
810 READ# 1 ; H,V.T1,B1,R1,L1,T2,B2,R2,L2.G$,Q1$,X1.X2,I2,Q2$,Y1.Y2.I4,I1,I3,
I6,C3,A1,S8
815 READ# 1 ; Q3$,Q4$,C7,A2,S7,X$,Q5$.Y$,Q6$,C6,A3.S6,T$,Q7$
820 ASSIGN# 1 TO *
821 R$="OLD"
822 IF D1#1 THEN GOTO 825
823 GOSUB 1575
824 GOSUB 1995
825 IF D1=3 THEN GOTO 315
830 GOSUB 1060
835 GOSUB 1575
840 GOSUB 1995
845 CLEAR
885 DISP "WHAT IS THE NAME OF THE FILE"
890 DISP "WHICH SPECIFIES THE DESIRED"
895 DISP "FORMAT?  INCLUDE STORAGE DEVICE-"
900 DISP "DESIGNATOR."
```

"GRAFIT" Program Listing--continued.

```
905 INPUT F$
910 ASSIGN#,1 TO F$
915 READ# 1 ; H,V,T1,B1,R1,L1,T2,B2,R2,L2,G$,Q1$,X1,X2,I3,Q2$,Y1,Y2,I4,I1,I5,I
I6,C8,A1,S8
920 READ# 1 ; Q3$,Q4$,C7,A2,S7,X$,Q5$,Y$,Q6$,C6,A3,S6,T$,Q7$
925 ASSIGN# 1 TO *
930 GOSUB 1060
935 GOSUB 1575
940 GOSUB 1995
945 ! PLOT SQUARE
950 IPLOT -S5*S1*I1*P2/(200*P1),S5*S2*I2/200,-2
955 IPLOT S5*S1*I1*P2/(100*P1),0,-1
960 IPLOT 0,-S5*S2*I2/100,-1
965 IPLOT -S5*S1*I1*P2/(100*P1),0,-1
970 IPLOT 0,S5*S2*I2/100,-1
975 IPLOT S5*S1*I1*P2/(200*P1),-S5*S2*I2/200,-2
980 RETURN
985 ! PLOT TRIANGLE
990 IPLOT -S5*S1*I1*P2/(200*P1),-S5*S2*I2/347,-2
995 IPLOT S5*S1*I1*P2/(100*P1),0,-1
1000 IPLOT -S5*S1*I1*P2/(200*P1),S5*S2*I2/115,-1
1005 IPLOT -S5*S1*I1*P2/(200*P1),-S5*S2*I2/115,-1
1010 IPLOT S5*S1*I1*P2/(200*P1),S5*S2*I2/347,-2
1015 RETURN
1020 ! PLOT DIAMOND
1025 IPLOT 0,-S5*S2*I2/141,-2
1030 IPLOT S5*S1*I1*P2/(141*P1),S5*S2*I2/141,-1
1035 IPLOT -S5*S1*I1*P2/(141*P1),S5*S2*I2/141,-1
1040 IPLOT -S5*S1*I1*P2/(141*P1),-S5*S2*I2/141,-1
1045 IPLOT S5*S1*I1*P2/(141*P1),-S5*S2*I2/141,-1
1050 IPLOT 0,S5*S2*I2/141,-2
1055 RETURN
1060 ! !!!!!!GRID PLOTTING!!!!!!
1061 PRINT H;V;T1;B1;R1;L1;T2;B2;R2;L2
1065 O1=IP(H*25.4-5)
1070 O2=IP(V*25.4-3)
1075 O3=IP(L1*25.4-5)
1080 O4=IP(B1*25.4-3)
1085 O5=IP(O1-R1*25.4)
1090 O6=IP(O2-T1*25.4)
1095 P1=O5-O3
1100 P2=O6-O4
1105 IF P1>P2 THEN GOTO 1135
1110 P3=IP(L2*2540/P1)
1115 P4=IP(B2*2540/P1)
1120 P5=100-IP(R2*2540/P1)
1125 P6=100/RATIO-IP(T2*2540/P1)
1130 GOTO 1155
1135 P3=IP(L2*2540/P2)
1140 P4=IP(B2*2540/P2)
1145 P5=RATIO*100-IP(R2*2540/P2)
1150 P6=100-IP(T2*2540/P2)
1151 PRINT O1;O2;O3;O4;O5;O6
1152 PRINT P1;P2;P3;P4;P5;P6
1155 LIMIT O3,O5,O4,O6
1160 LOCATE P3,P5,P4,P6
1165 IF G$="N" THEN GOTO 1175
1170 FRAME
1175 IF Q1$="N" THEN GOTO 1190
1176 IF R$="OLD" THEN GOTO 1190
```

"GRAFIT" Program Listing--continued.

```
1180 X1=LOG(X1)/LOG(10)
1185 X2=LOG(X2)/LOG(10)
1186 I4=LOG(I4)/LOG(10)
1190 IF Q2$="N" THEN GOTO 1200
1191 IF R$="CLD" THEN GOTO 1200
1195 Y1=LOG(Y1)/LOG(10)
1196 Y2=LOG(Y2)/LOG(10)
1197 I3=LOG(I3)/LOG(10)
1200 S1=(X2-X1)/I1
1205 S2=(Y2-Y1)/I2
1210 DEG
1215 SCALE X1,X2,Y1,Y2
1220 CSIZE C8.A1.S8
1225 AXES S1,S2.I4.I3.I5,I6
1230 IF Q3$="A" THEN GOTO 1310
1235 LORG 6
1240 FOR X=X1 TO X2 STEP S1
1245 IF Q1$="N" THEN GOTO 1285
1250 MOVE X,Y1-C8/120*S2*I2
1255 LABEL "10"
1260 LORG 3
1265 MOVE X+A1*C8/60*S1*I1/RATIO.Y1-C8/200*S2*I2
1266 IF X<0 THEN IMOVE .67*A1*C8/60*S1*I1/RATIO.0
1270 CSIZE .67*C8,A1,S8
1275 LABEL IP(X)
1280 GOTO 1290
1285 MOVE X,Y1-C8/200*S2*I2
1286 LABEL X
1290 CSIZE C8,A1.S8
1295 LORG 6
1300 NEXT X
1305 GOTO 1380
1310 LORG 4
1315 FOR X=X1 TO X2 STEP S1
1320 IF Q1$="N" THEN GOTO 1360
1325 MOVE X.Y2+C8/120*S2*I2
1330 LABEL "10"
1335 LORG 1
1340 MOVE X+A1*C8/60*S1*I1/RATIO,Y1+C8/200*S2*I2
1341 IF X<0 THEN IMOVE .67*A1*C8/60*S1*I1/RATIO,0
1345 CSIZE .67*C8.A1.S8
1350 LABEL IP(X)
1355 GOTO 1365
1360 MOVE X,Y2+C8/200*S2*I2
1361 LABEL X
1365 CSIZE C8,A1,S8
1370 LORG 4
1375 NEXT X
1380 IF Q4$="R" THEN GOTO 1415
1385 LORG 8
1390 FOR Y=Y1 TO Y2 STEP S2
1391 IF Q2$="N" THEN GOTO 1400
1392 MOVE X1-C8*A1/60*S1*I1/RATIO.Y
1393 LABEL "10"
1394 LORG 1
1395 IMOVE A1*C8/60*S1*I1/RATIO.C8/200*S2*I2
1396 IF Y<0 THEN IMOVE .67*A1*C8/60*S1*I1/RATIO.0
1397 CSIZE .67*C8.A1,S8
1398 LABEL IP(Y)
1399 GOTO 1402
```

"GRAFIT" Program Listing--continued.

```
1400 MOVE X1-C8*A1/266*S1*I1/RATIO,Y
1401 LABEL Y
1402 CSIZE C8,A1,S8
1403 LORG 8
1405 NEXT Y
1410 GOTO 1440
1415 LORG 2
1420 FOR Y=Y1 TO Y2 STEP S2
1425 MOVE X2+C8/100*S2*I2,Y
1430 LABEL Y
1435 NEXT Y
1440 CSIZE C7,A2,S7
1445 IF Q5$="A" THEN GOTO 1465
1450 LORG 6
1455 MOVE X1+S1*I1/2,Y1-C8/33*S2*I2
1460 GOTO 1475
1465 LORG 4
1470 MOVE X1+S1*I1/2,Y2+C8/33*S2*I2
1475 LABEL X$
1480 IF Q6$="R" THEN GOTO 1505
1485 LORG 6
1490 LDIR 90
1495 MOVE X1-S1*I1*L2*25.3/F1,Y1+S2*I2/2
1500 GOTO 1520
1505 LORG 4
1510 LDIR 90
1515 MOVE X2+S1*I1*R*25.3/F1,Y1+S2*I2/2
1520 LABEL Y$
1525 CSIZE C6,A3,S6
1530 LDIR 0
1535 IF Q7$="B" THEN GOTO 1555
1540 LORG 6
1545 MOVE X1+S1*I1/2,Y2+S2*I2*T2*25.3/F2
1550 GOTO 1565
1555 LORG 4
1560 MOVE X2=S1*I1/2,Y1-S2*I2*B*25.3/F2
1565 LABEL T$
1570 RETURN
1575 ! !!!!DATA PLOTTING!!!!!!
1580 CLEAR
1585 DISP "*********PLOTTING DATA**********"
1590 DISP
1595 DISP "      YOU MAY ENTER PLOTTING DATA IN THREE WAYS:"
1600 DISP "      1.  ENTER X,Y PAIRS INDIVIDUALLY."
1605 DISP "      2.  CREATE ASUBROUTINE TO GENERATE Y AS A FUNCTION OF X."
1610 DISP "      3.  READ X,Y DATA FROM A DATA FILE."
1615 DISP " WHICH WAY WILL THE DATA BE ENTERED?  (1,2 OR 3?)"
1620 INPUT N9
1625 CLEAR
1630 DISP "INTERCONNECTING LINETYPE?  (0-8)"
1635 INPUT T3
1640 DISP "SYMBOL AT DATA POINT?"
1645 DISP "  0  NONE"
1650 DISP "  1  SQUARE"
1655 DISP "  2  TRIANGLE"
1660 DISP "  3  DIAMOND"
1665 INPUT S4
1670 IF S4=0 THEN GOTO 1685
1675 DISP "SYMBOL SIZE - % OF GRID HT"
1680 INPUT S5
```

"GRAFIT" Program Listing--continued.

```
1685 ON N9 GOTO 1690,1755,1910
1690 DISP "NUMBER OF X,Y PAIRS?"
1695 INPUT N0
1700 IF T3=0 THEN GOTO 1710
1705 LINETYPE T3
1710 FOR N=1 TO N0
1715 DISP "X COORDINATE?"
1720 INPUT X
1725 DISP "Y COORDINATE?"
1730 INPUT Y
1735 GOSUB 1995
1740 NEXT N
1745 PENUP
1750 RETURN
1755 CLEAR
1760 DISP "  SUBROUTINE [illegible]"
1765 DISP "     YOU MAY PLOT UP TO FOUR DIFFERENT FUNCTIONS.  TO"
1770 DISP "DO THIS, YOU MUST FIRST DEFINE THE FUNCTIONS IN SUBROUTINES"
1775 DISP "     FUNCTION 1 - LINE 6000"
1780 DISP "     FUNCTION 2 - LINE 7000"
1785 DISP "     FUNCTION 3 - LINE 8000"
1790 DISP "     FUNCTION 4 - LINE 9000"
1795 DISP
1800 DISP "     HAVE YOU DEFINED THE FUNC-"
1805 DISP "     TIONS? (Y OR N)"
1810 INPUT Q9$
1815 IF Q9$="Y" THEN GOTO 1835
1820 CLEAR
1825 DISP "CREATE FUNCTIONS, THEN RERUN PROGRAM"
1830 GOTO 2[illegible]
1835 DISP "WHICH FUNCTION DO YOU WANT TO PLOT?  (1,2,3 OR 4)"
1840 INPUT F1
1845 IF F1#1 THEN GOTO 1850
1847 GOSUB 6000
1848 GOSUB 1995
1850 DISP "HOW MANY INTERVALS ALONG X"
1855 DISP "AXIS?"
1860 INPUT I1
1865 S=(X2-X1)/I1
1870 IF T3=0 THEN GOTO 1880
1875 LINETYPE T3
1880 FOR X=X1 TO X2 STEP S
1885 ON F1 GOSUB 6000,7000,8000,9000
1890 GOSUB 1995
1895 NEXT X
1900 PENUP
1905 RETURN
1910 DISP "WHAT IS THE NAME OF THE DATA FILE?  (INCLUDE STORAGE DEVICE CODE
 - e.g. :T)"
1915 INPUT N$
1920 ASSIGN# 1 TO N$
1925 READ# 1 : N0,C1,C2,R3,R4,R5
1926 PRINT "CO=";C1,"CF=";C2,"RF=";R1,"RR=";R2,"PS=";R3
1930 FOR N=1 TO N0
1935 READ# 1 : X(N),Y(N),F(N)
1940 NEXT N
1945 ASSIGN# 1 TO *
1950 IF T3=0 THEN GOTO 1960
1955 LINETYPE T3
1960 M=0
```

"GRAFIT" Program Listing--continued.

```
1960 FOR N=1 TO N3
1965 X=J(N)
1970 Y=K(N)
1971 IF N<50*M THEN GOTO 1979
1972 CSIZE 4,.5,0
1973 LORG 5
1974 LABEL "+"
1975 M=M+1
1976 IF O1$="Y" THEN X=LOG(X)/LOG(10)
1977 IF O2$="Y" THEN Y=LOG(Y)/LOG(10)
1978 MOVE X,Y
1979 GOSUB 2195
1980 NEXT N
1985 PENUP
1990 RETURN
1995 ! !!!!!!WRAPUP!!!!!!!
2000 CLEAR
2005 DISP "PLOT ANOTHER DATA SET?  (Y OR N)"
2010 INPUT O9$
2015 IF O9$="N" THEN GOTO 2030
2020 GOSUB 1575
2025 GOSUB 1995
2030 DISP "ANOTHER PLOT?  (Y OR N)"
2035 INPUT O9$
2040 DISP "SAVE THESE PLOT PARAMETERS AS A PERMANENT FORMAT?  (Y OR N)"
2045 INPUT P1$
2050 IF P1$="N" THEN GOTO 2125
2055 DISP "WHAT SHOULD THE PLOT PARAMETER FILE BE NAMED?  INCLUDE STORAGE [illegible]
 DESIGNATOR."
2060 INPUT P$
2065 ON ERROR GOTO 2080
2070 CREATE P$,10
2075 GOTO 2100
2080 IF ERRN=63 THEN CLEAR
2085 DISP "FILE NAME, ",P$,"ALREADY EXISTS..."
2090 OFF ERROR
2095 GOTO 2055
2100 ASSIGN# 1 TO P$
2105 PRINT# 1 : H,V,T1,B1,R1,L1,T2,B2,R2,L2,G$,O1$,X1,X2,I3,O2$,[illegible]
2,I6,C8,A1,S8
2110 PRINT# 1 : O3$,O4$,C7,A2,S7,X$,O5$,Y$,O6$,C6,A3,S6,T$,O7$
2115 ASSIGN# 1 TO *
2120 GOTO 2175
2125 ON ERROR GOTO 2140
2130 CREATE "PLTDAT:D700",10
2135 GOTO 2155
2140 IF ERRN=63 THEN PURGE "PLTDAT:D700"
2141 IF ERRN#63 THEN PRINT "ERRN= ";ERRN
2145 OFF ERROR
2150 GOTO 2130
2155 ASSIGN# 1 TO "PLTDAT:D700"
2160 PRINT# 1 ; H,V,T1,B1,R1,L1,T2,B2,R2,L2,G$,O1$,X1,X2,I3,O2$,Y1,Y2,I4,[illegible]
2,I6,C8,A1,S8
2165 PRINT# 1 : O3$,O4$,C7,A2,S7,X$,O5$,Y$,O6$,C6,A3,S6,T$,O7$
2170 ASSIGN# 1 TO *
2175 IF O9$="Y" THEN GOTO 110
2180 CLEAR
2185 DISP "DONE"
2190 END
2195 ! !!!!!POINT PLOT!!!!!!!
```

"GRAFIT" Program Listing--continued.

```
2200 IF Q1$="Y" THEN X=LOG(X)/LOG(10)
2205 IF Q2$="Y" THEN Y=LOG(Y)/LOG(10)
2210 IF T3=0 THEN GOTO 2230
2215 IF N=1 THEN GOTO 2230
2220 PLOT X,Y,-1
2225 GOTO 2235
2230 PLOT X,Y
2235 IF S4=0 THEN GOTO 2250
2240 LINETYPE 1
2245 ON S4 GOSUB 945,985,1020
2250 RETURN
6000 DISP "C1?"
6010 INPUT C1
6020 DISP "C2?"
6030 INPUT C2
6040 DISP "R3?"
6050 INPUT R3
6060 DISP "R4?"
6070 INPUT R4
6080 DISP "R5?"
6090 INPUT R5
6100 CLEAR
6110 DISP "HOW MANY DATA POINTS?"
6120 INPUT N3
6121 M=1
6130 FOR N=1 TO N3
6140 GOSUB 6500
6141 IF N<50*M THEN GOTO 6150
6142 CSIZE 4,.5,0
6143 LORG 5
6144 LABEL "+"
6145 M=M+1
6146 MOVE X,Y
6150 GOSUB 2195
6160 NEXT N
6165 PENUP
6166 PRINT "CD=";C1,"CF=";C2,"RF=";R3,"RF=";R4,"TS=";R5
6170 DISP "SAVE THIS DATA FILE? (Y OR N)"
6180 INPUT Z$
6190 IF Z$="N" THEN RETURN
6200 DISP "NAME OF DATA FILE?"
6210 INPUT N$
6220 ON ERROR GOTO 6250
6230 CREATE N$,30
6240 GOTO 6310
6250 CLEAR
6260 IF ERRN=63 THEN GOTO 6290
6270 DISP "ERROR # = ";ERRN
6280 GOTO 6370
6290 DISP "FILE NAME, ";N$;" ALREADY EXISTS..."
6300 GOTO 6200
6310 ASSIGN# 1 TO N$
6320 PRINT# 1 ; N3,C1,C2,R1,R2,R3
6330 FOR N=1 TO N3
6340 PRINT# 1 ; I(N),J(N),K(N)
6350 NEXT N
6360 ASSIGN# 1 TO *
6370 OFF ERROR
6380 RETURN
6500 I(N)=2*P1*10^(-.602+.02*(N-1))
```

"GRAFIT" Program Listing--continued.

```
6505 D1=1+(I(N)*C1*R3)^2
6510 Z1=R3/D1
6515 Z2=-I(N)*C1*R3^2/D1
6520 D2=1+(I(N)*C2*R4)^2
6525 Z3=R4/D2
6530 Z4=-I(N)*C2*R4^2/D2
6535 Z5=R5+Z1+Z3
6545 Z6=Z2+Z4
6550 Z7=SQR(Z5^2+Z6^2)
6555 Z8=ATN(Z6/Z5)
6560 J(N)=I(N)/(2*PI)
6565 K(N)=Z8
6566 X=J(N)
6567 Y=K(N)
6570 RETURN
7000 ! !!!!!FUNCTION# 2!!!!!!!
7015 Y=X^2
7020 RETURN
7025 ! !!!!!FUNCTION# 3!!!!!!!
8000 ! !!!!!FUNCTION# 3!!!!!!!
8030 Y=X^3
8035 RETURN
9000 ! !!!!!FUNCTION# 4!!!!!!!
9045 Y=6*ABS(SIN(X*180))
9050 RETURN
```

BIBLIOGRAPHY

1. I. Epelboin, M. Keddam and H. Takenouti, J. Appl. Electrochem., vol. 2, pg. 71, 1972.

2. I. Epelboin, P. Morel, and H. Takenouti, J. Electrochem. Soc., vol. 118, pg. 1282, 1971.

3. I Epelboin, C. Gabrielli, M. Keddam and L. Raillon, J. Electroanal. Chem., vol. 93, pg. 155, 1975.

4. G. Blanc, I. Epelboin, C. Gabrielli and M. Keddam, Electrochim. Acta, vol. 20, pg. 599, 1975.

5. S. C. Creason and D. E. Smith, J. Electro. Anal. Chem., vol. 36, pg. A1, 1972.

6. S. C. Creason and D. E. Smith, Ibid., vol. 40, pg. 1, 1972.

7. S. C. Creason, J. W. Hayes and D. E. Smith, Ibid., vol. 47, pg. 9, 1973.

8. S. C. Creason and D. E. Smith, Anal. Chem., vol. 45, no. 14, pg. 2401, 1973.

9. D. E. Smith, Ibid., vol. 48, no. 2, pg. 221A, 1976.

10. D. E. Smith, Ibid., vol. 48, no. 6, pg. 517A, 1976.

11. W. J. Lorenz and F. Mansfeld, to be published.

12. R.J. Schwall, A. M. Bond, R. J. Lloyd, J. G. Larsen and D. E. Smith, Ibid., vol. 49, no. 12, pg. 1797, 1977.

13. R. DeLevie, J. W. Thomas and K. M. Abbey, J. Electroanal. Chem., vol. 62, pg. 111, 1975.

14. U. Bertocci, J. Electrochem. Soc., vol. 127, pg. 1931, 1980.

15. H. L. von Helmholtz, Wied. Ann., vol. 7, pg. 337, 1879.

16. J. O. M. Bockris and A. K. N. Reddy. Modern Electrochemistry, vol. 2, Plenum Press, New York, 1970.

17. G. Gouy, J. Chim. Phys. (Paris), vol. 29, no. 7, pg. 145, 1903.

18. D. L. Chapman, Phil. Mag., vol. 25, no. 6, pg. 475, 1913.

19. O. Stern, Z. Elektrochem., vol. 30, pg. 508, 1924.

20. J. A. V. Butler, Trans. Faraday Soc., vol. 28, pg. 379, 1932.

21. H. Gerischer and K. J. Vetter, Z. Physik. Chem., vol. 197, pg. 82, 1951.

22. E. Warburg, Wied. Ann., vol. 67, pg. 493, 1899; Drud. Ann., vol. 6, pg. 125, 1901.

23. H. Gerischer, Z. Physik. Chem., vol. 198, pg. 286, 1951; vol. 201, pg. 55, 1952.

24. W. Lorenz, Z. Physik. Chem., vol. 202, pg. 275, 1953; Naturwiss., vol. 40, pg. 576, 1953.

25. K. J. Vetter, Electrochemical Kinetics, pp. 327-328, Academic Press, New York, 1967.

26. Ibid., pp. 345-347.

27. D. C. Grahame, J. Electrochem. Soc., vol. 99, pg. 370C, 1952.

28. E. B. Randles, Disc. Faraday Soc., vol. 1, pg. 11, 1947.

29. K. J. Vetter, Ibid., ppg. 396-454, 1967.

30. M. J. Pryor, Z. Elektrochem., vol. 62, pg. 782, 1958.

31. A. F. Beck, M. A. Heine, D. S. Keir, I vol Rooyan, and M. J. Pryor, Corrosion Science, vol. 2, pg. 133, 1962.

32. A. F. Beck, M. A. Heine, E. J. Caule and M. J. Pryor, Corrosion Science, vol. 7, pg. 1, 1967.

33. M. A. Heine, D. S. Keir, and M. J. Pryor, J. Electrochem. Soc., vol. 110, pg. 1205, 1963.

34. M. A. Heine, D. S. Keir, and M. J. Pryor, Ibid., vol. 112, pg. 24, 1965.

35. M. A. Heine and M. J. Pryor, J. Electrochem. Soc., vol. 114, pg. 1001, 1967.

36. J. A. Richardson and G. C. Wood, J. Electrochem. Soc., vol. 120, pg. 193, 1973.

37. J. A. Richardson, G. C. Wood and A. J. Breen, Thin Solid Films, vol. 16, pg. 81, 1973.

38. S. Haruyama and T. Tsuru in "Passivity of Metals," R. P. Frankenthal and J. Kruger, Editors, pg. 564, NACE, Houston, 1978.

39. C. Wagner and W. Traud, Z. Elektrochem., vol. 44, pg. 391, 1938.

40. M. Stern and A. L. Geary, J. Electrochem. Soc., vol. 102, pg. 609, 1955.

41. M. Stern and A. L. Geary, Ibid., vol. 104, pg. 56, 1957.

42. M. Stern, Corrosion, vol. 14, pg. 60, 1958.

43. L. M. Callow, J. A. Richardson and J. L. Dawson, Br. Corros. J., vol. 11, no. 3, pg. 123, 1976.

44. M. Sluyters-Rehbach and J. H. Sluyters in Electroanalytical Chemistry, vol. 4, A. J. Bard, Editor, Marcel Dekker, New York, 1970.

45. D. E. Smith in Computers in Chemistry and Instrumentation, Vol. 2, J. S. Mattson, H. B. Mark and H. C. MacDonald, Editors, Marcel Dekker, New York, 1972.

46. C. T. Chen., Introduction to Linear System Theory, Holt, Rinehart and Winston, New York, 1970.

47. R. W. Schideler and U. Bertocci, J. Res. Nat. Bur. Stand., vol. 85, pg. 211, 1980.

48. A. A. Pilla, J. Electrochem. Soc., vol. 117, no. 4, pg. 467, 1970.

49. J. S. Bendat and A. G. Piersol, Measurement and Analysis of Random Data, John Wiley, New York, 1966.

50. R. L. Birke, Anal. Chem., vol. 43, pg. 1253, 1971.

51. K. Doblhofer and A. A. Pilla, J. Electro. Anal. Chem., vol. 39, pg. 91, 1972.

52. J. W. Cooley and J. W. Tukey, Math. Comp., vol. 19, pg. 297, 1965.

53. J. W. Hartwell, IBM J. Res. Develop., vol. 15, pg. 355, 1971.

54. W. H. Smyrl and S. L. Pohlman, "Determination of Corrosion Parameters by Digital Impedance Analysis," SAND-78-0038C, Sandia Labs, Albuquerque, New Mexico, 1978.

55. Private communication with P. M. Russell, Dow Chemical Corp., Midland Michigan, March 1983.

56. SRP G5-72, 1977 Annual Book of ASTM Standards, Part 10, American Society for Testing and Materials, 1977.

57. Transmission Systems for Communications, Fourth Edition, Bell Telephone Laboratories, Inc., 1970.

58. Hewlett Packard Application Note 240-0, "Digital Signal Analysis-Time and Frequency Domain Measurements."

59. Operator's Manual, HP5420A, Chapter 5, Hewlett Packard Corporation.

BIOGRAPHICAL SKETCH

Joseph Warren Hager was born August 17, 1946, in Morristown, New Jersey. He was raised and educated through high school in rural New Jersey graduating fourth in a class of 360 at Hunterdon Central High School in 1964. Interrupting his undergraduate study of engineering at Purdue University, he spent two years serving as a missionary for the Church of Jesus Christ of Latter-Day Saints (Mormon) in the Republic of Austria. He resumed his studies in 1969 and completed the Bachelor of Science in Chemical Engineering degree in 1971 and a Master of Science in Engineering in 1972. From 1972 to 1973, he conducted an independent research project using holographic interferometry at the Technische Hochschule in Darmstadt, Federal Republic of Germany, under the auspices of a Fulbright-Hayes Scholarship. Having been commissioned an Air Force Second Lieutenant through the Reserve Officer Training Corps program at Purdue, he then began active duty as a project engineer in the Manufacturing Technology Division of the Air Force Materials Laboratory in Dayton, Ohio. Prior to reentering graduate school in 1977, he spent two years as the materials liaison engineer to the F-16 Air Combat Fighter System Program Office. Upon completion of course work and residency requirements at the University of Florida, Captain Hager undertook the research portion of his doctoral studies at the Solar Energy Research Institute in Golden, Colorado. In 1980, he was assigned to the United States Air Force Academy where he is currently an Assistant Professor in the Department of

Engineering Mechanics. Captain Hager is married to the former MarJean Johnson of Rexburg Idaho. They have a son, Aric, and two daughters, Kirsten and Anneke.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Ellis Verink, Chairman
Professor of Materials Science and Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Rolf Hummel
Professor of Materials Science and Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

John Ambrose
Associate Professor of Materials Science and Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Gerhard Schmid
Associate Professor of Chemistry

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate Council, and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

December 1983

Herbert A. Bevis

Dean, College of Engineering

Dean, Graduate School